Algorithmic Term Rewriting Systems

Isihara, A.

2010

document version
Publisher's PDF, also known as Version of record

Algorithmic Term Rewriting Systems

Ariya Isihara

Printed and bound by Wöhrmann Print Service
Published by VU University Press
ISBN: 978-90-8659-442-9

Cover design by Ariya Isihara

Front Cover:	"The Creation of Infinity" Michelangero Buonarroti – Ariya Isihara
Back Cover:	"The Creation of Natural Numbers" Michelangero Buonarroti – Ariya Isihara
Invitation Card:	"Annunciation of Infinity" El Greco – Ariya Isihara

VRIJE UNIVERSITEIT

Algorithmic Term Rewriting Systems

ACADEMISCH PROEFSCHRIFT

ter verkrijging van de graad Doctor aan
de Vrije Universiteit Amsterdam,
op gezag van de rector magnificus
prof.dr. L.M. Bouter,
in het openbaar te verdedigen
ten overstaan van de promotiecommissie
van de faculteit der Exacte Wetenschappen
op donderdag 25 maart 2010 om 15.45 uur
in de aula van de universiteit,
De Boelelaan 1105

door
Ariya Isihara
geboren te Nagoya, Aichi, Japan

promotor:	prof.dr. J.W. Klop
copromotor:	dr. R.C. de Vrijer

Acknowledgement

I would like to follow the classical custom to declare that this thesis is organized and written all by myself. But, the thesis would obviously have never been finished without help from many people I am acquainted with, or things that have made me what I am.

I learned analytic number theory in my bachelor years, supervised by Yoshio Tanigawa. I owe him a lot; his words and his smile in my memory have been always encouraging to me.

Then, I switched my major to computer science in my Master course at Kyoto University. I would like to thank Masahito Hasegawa, my supervisor there. During the Master course, he has been an expert guide for me. My acknowlegement includes the other professors, Jacques Garrigue and Susumu Nishimura. I have learned from them the basis of computer science. In addition, I owed the enjoyable atmosphere there to my colleagues: Hidehisa Arikawa, Kazuyuki Asada, Naoya Enomoto, Yuichiro Hoshi, Hiraku Kawanoue, Sin-ya Katsumata, Keiko Nakata, Takeshi Nozawa, and Hisanori Ohashi.

For obtaining my present position, I am grateful to Jan Willem Klop and Roel de Vrijer for having invited me to Amsterdam. Repeating the same names, I would acknowledge my supervisor Roel de Vrijer and my supervisor Jan Willem Klop. It has been the most pleasant stage of researches to show an idea to Jan Willem. Roel has encouraged me very much when I was in the most unpleasant stage of researches, putting my results on paper. Here I would quote his proverb: the best definition is no definition.* That considerably reduced my uncomfortability in the writing process. Unfortunately the thesis contains a lot of Definitions, however, the paper I dedicated to him [28] contains no definition, in appreciation of his proverb. My acknowledgement also includes the other colleagues for an excellent atmosphere at the university: Rena Bakhshi, Taolue Chen, Jörg Endrullis, Wan Fokkink, Clemens Grabmayer, Helle Hvid Hansen, Dimitri Hendriks, Cynthia Kop, Femke van Raamsdonk, Jan Rutten. Jöerg, Clemens, Dimitri, and Jan Willem are the first coauthors I have ever had. The work we did together underlies many ideas in this thesis.

Special thanks go to Ichiro Hasuo; he studied at Radboud Universiteit Nijmegen when I came to Amsterdam, and he showed me how a Japanese man adapts in Holland.

*Paraphrasing Paul Halmos [24]: the best notation is no notation.

As to my leisure life in Holland, I am grateful to the late Professional Go Player Kaoru Iwamoto for the present prevalence of the European Go society, and I thank the members of the Go clubs 'Twee Ogen' and 'Goog'. Especially, very special thanks go to Judith van Dam and Harry Weerheijm for hosting some poker parties, which were really enjoyable.

I am grateful to the members of the reading committee: Stefan Blom, Jan Rutten, Andreas Weiermann, and Hans Zantema for their reviews. Nevertheless, the remaining errors of the thesis are all mine.

Last, but not least, I am grateful to my family, especially my wife Keiko and my son Haruka. Thanks to them, I have been able to keep my sanity; to be precise, I was always eventually able to return to sanity.

Ariya Isihara
Amsterdam, January 2010

Contents

Chapter 0

Introduction

In theoretical computer science, in particular the area of algebraic specification of abstract data types, sorted specifications based on the principle of induction are well-known. Here one is concerned with finite data such as natural numbers, booleans, finite trees, finite lists, and so on. More recently, a 'dual' specification method has become prominent: that of coalgebraic specifications of infinite data, also called codata. Typical codata are lazy natural numbers, infinite trees, infinite lists or streams. Whereas in the world of finite data, induction is the salient principle of definition and proof, it is replaced by coinduction in the realm of codata.

In this thesis we develop a general framework that extends both the inductive and coinductive specifications: the framework of *algorithmic term rewriting systems*. The class of algorithmic term rewriting systems provides a scheme for function specifications employing both inductive and coinductive constructions. When a function specification is given, we are concerned whether the specification is well-defined or not. This leads to the primary desired property of algorithmic term rewriting systems: all the specifications that can be given as expressions in the system should be well-defined. We shall call this property *'productivity'*. This description of productivity is still very informal. A contribution of this thesis is giving it a technically precise interpretation. The resulting notion of productivity of an algorithmic system is fundamental, on a par with the notions of termination and confluence for (finitary) term rewriting systems.

Productivity turns out to be the consequence of three secondary properties. First, *infinitary normalization* (WN) guarantees that an expression has a possibly infinite normal form. Secondly, *domain normalization* (DN) guarantees that the normal form is within the intended domain of results. Thirdly, *constructor normalization* (CN) guarantees that the normal form is built solely from constructors without defined function symbol. We give conditions for each of the three properties WN, DN, CN, and in some instances even characterizations. Together, they form conditions that ensure productivity of the algorithmic term rewriting system.

As an application of the theory developed here, we consider in the final chapter a fairly complicated infinite data type known as tree ordinals.

These are important in the theory of ordinal notations in proof theory, a branch of mathematical logic. They also embody a study of the expressivity of the first order term rewriting framework, and we show that this expressivity is large: we can express ordinals far larger than the Feferman–Schütte ordinal known as Γ_0.

In order to set the stage, in this chapter we show how the framework of algorithmic systems and the notion of productivity arise. This chapter gives an informal introduction; later the notions will be formally introduced.

0.1 The concept of productivity

Let us start by dealing with finitary objects. We think of representing natural numbers as a paradigmatic example. The set of natural numbers $\mathbf{N}$ can be inductively specified as follows:

1. 0 is a natural number.
2. If n is a natural number, then so is the *successor* $n + 1$.

So, we employ the *constructors*

$$\begin{aligned} 0 :&\quad () \to \mathsf{NAT} \\ s :&\ \mathsf{NAT} \to \mathsf{NAT} \end{aligned}$$

to represent natural numbers, where NAT denotes the set of terms (representations) of natural numbers. We call such objects as NAT constructed by inductive means *'inductive objects'*.

We have the *representing function* $\ulcorner - \urcorner : \mathbf{N} \to \mathsf{NAT}$ inductively defined by

$$\begin{aligned} \ulcorner 0 \urcorner &\equiv 0 \\ \ulcorner n + 1 \urcorner &\equiv s \ulcorner n \urcorner \end{aligned}$$

and the *semantics function* $[\![-]\!] : \mathsf{NAT} \to \mathbf{N}$ inductively given by

$$\begin{aligned} [\![0]\!] &= 0 \\ [\![s\,t]\!] &= [\![t]\!] + 1 \end{aligned}$$

We use $\equiv$ to denote syntactic identity of terms, since the symbol $=$ usually stands for the equivalence relation generated by the reduction relation. Figure 1 illustrates those functions.

Next, we consider defining functions on natural numbers; addition and multiplication, for example. Given natural numbers n and m, both the sum

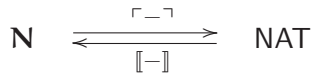


Figure 1: *Representation of* $\mathbf{N}$.

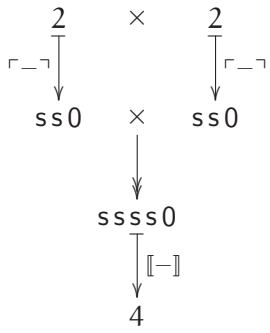


Figure 2: *Two times two is four.*

$n + m$ and the product $n \times m$ can be defined by mathematical induction on m as follows:

$$\begin{aligned} n + 0 &= n & n \times 0 &= 0 \\ n + (m + 1) &= (n + m) + 1 & n \times (m + 1) &= (n \times m) + n \end{aligned}$$

From this definition, the following (finitary) term rewriting system arises:

$$\begin{aligned} n + 0 &\to n & n \times 0 &\to 0 \\ n + \mathsf{s}\, m &\to \mathsf{s}(n + m) & n \times \mathsf{s}\, m &\to (n \times m) + n \end{aligned}$$

For example, to compute two times two, we perform reductions

$$\begin{aligned} \mathsf{ss0} \times \mathsf{ss0} &\to (\mathsf{ss0} \times \mathsf{s0}) + \mathsf{ss0} \\ &\to \mathsf{s}((\mathsf{ss0} \times \mathsf{s0}) + \mathsf{s0}) \\ &\to \mathsf{ss}((\mathsf{ss0} \times \mathsf{s0}) + 0) \\ &\to \mathsf{ss}(\mathsf{ss0} \times \mathsf{s0}) \\ &\to \mathsf{ss}((\mathsf{ss0} \times 0) + \mathsf{ss0}) \\ &\to \mathsf{sss}((\mathsf{ss0} \times 0) + \mathsf{s0}) \\ &\to \mathsf{ssss}((\mathsf{ss0} \times 0) + 0) \\ &\to \mathsf{ssss}(\mathsf{ss0} \times 0) \\ &\to \mathsf{ssss0} \end{aligned}$$

In the above computation we deal with the leftmost appearance of redexes (*red*ucible *ex*pressions) in each step. However, regardless of the choice of a redex, we reach the unique result $\mathsf{ssss0}$ after finitely many steps. We write $\twoheadrightarrow$ to denote the reflexive transitive closure of the single-step reduction $\to$ (e.g. $\mathsf{ss0} \times \mathsf{ss0} \twoheadrightarrow \mathsf{ssss0}$). Thus, we conclude $2 \times 2 = 4$ (see Figure 2). It should be noticed that the result $\mathsf{ssss0}$ contains no $+$ or $\times$; these defined symbols are eliminated by the reductions.

To be a little more formal, we call the expressions generated by means of $\mathsf{0}$ and s, *'constructor normal forms'*, and those generated by means of $\mathsf{0}$, s, $+$, and $\times$, *'terms'*. An expression 2×2 on natural numbers is translated to the term $\mathsf{ss0} \times \mathsf{ss0}$, and then computed to the value $\mathsf{ssss0}$, and finally translated back to the natural number 4.

In this example, it is not so difficult to observe that any term t can be reduced to a unique constructor normal form s by finitely many steps of reductions. Denote that s by $\mathrm{cnf}(\mathrm{t})$. Then, the binary function

$$+ : \mathbf{N} \times \mathbf{N} \to \mathbf{N}$$

specified by the function symbol $+$, is *well-defined* as

$$\mathfrak{n} + \mathfrak{m} = [\![\mathrm{cnf}(\ulcorner \mathfrak{n} \urcorner + \ulcorner \mathfrak{m} \urcorner)]\!]$$

and analogously for $\times$. Thus, a term rewriting system satisfying a certain property gives a definition of functions. So, we introduce the notion of productivity as the property that a term rewriting system should satisfy to define functions.

Concept 1 A term rewriting system is *productive* if the system defines some functions in the way discussed above. ◻

In this thesis we are concerned with productivity of term rewriting systems. In order to formulate the productivity property following the above concept, we distinguish defined function symbols from constructor function symbols.

Concept 2

1. A *constructor* symbol is a function symbol such as $\mathsf{0}$ and s, which is used to construct objects.

2. A *defined* symbol is a function symbol such as $+$ and $\times$, which is used to express operations on objects. ◻

Following the above concept, given a term rewriting system, we can formulate this division of function symbols as follows: a function symbol is a defined symbol if it appears at the root of a lefthand-side of some rules; a constructor symbol otherwise. Accordingly, we call terms containing only constructor symbols 'constructor normal forms'. Note that every constructor normal form is a normal form.

In addition, we call terms containing no variable symbol 'ground terms'. Clearly, every constructor normal form is ground.

Now we can formulate the productivity property, for finitary term rewriting systems, as follows.

Concept 3 A finitary term rewriting system is *productive* if, for every well-sorted ground term t, there exists a unique constructor normal form s such that $t \twoheadrightarrow s$. ⊡

0.2 The concept of properness

With the system of natural numbers constructed as in the previous section, we next consider constructing streams (infinite sequences) of natural numbers. Generally, given a set A, the set of streams of elements in A is coinductively given as follows:

> If x is a stream, then x is of the form $a : x'$, where $a \in A$ and x' is a stream.

In order to deal with these objects in the framework of term rewriting systems, we introduce infinite terms and transfinite reductions. For example, consider the function nats with the rules

$$\mathsf{nats}(n) \to n : \mathsf{nats}(\mathsf{s}\, n)$$

where ':' is a constructor of type $\mathsf{NAT} \times \mathsf{STREAM}_{\mathsf{NAT}} \to \mathsf{STREAM}_{\mathsf{NAT}}$, constructing a stream. Contrasting to inductive objects, we call such objects *'coinductive objects'*.

Starting from the term $\mathsf{nats}(0)$, we have a non-terminating reduction

$$\begin{aligned}\mathsf{nats}(0) &\to 0 : \mathsf{nats}(\mathsf{s}\,0)\\ &\to 0 : \mathsf{s} : \mathsf{nats}(\mathsf{s}\,\mathsf{s}\,0)\\ &\to 0 : \mathsf{s}\,0 : \mathsf{s}\,\mathsf{s}\,0 : \mathsf{nats}(\mathsf{s}\,\mathsf{s}\,\mathsf{s}\,0)\\ &\to \dots\end{aligned}$$

In this sequence of reductions, more and more reductions will keep a larger part of the term fixed. In other words, the position of the contracted redex gets deeper and deeper. Thus, we may as well say that the term $\mathsf{nats}(0)$ represents the limit of this infinite reduction sequence, that is

$$0 : \mathsf{s}\,0 : \mathsf{s}\,\mathsf{s}\,0 : \mathsf{s}\,\mathsf{s}\,\mathsf{s}\,0 : \mathsf{s}\,\mathsf{s}\,\mathsf{s}\,\mathsf{s}\,0 : \mathsf{s}\,\mathsf{s}\,\mathsf{s}\,\mathsf{s}\,\mathsf{s}\,0 : \dots$$

So, the notion of transfinite reduction naturally arises, which will be formalized in Definition 15. We use $\twoheadrightarrow$ to denote (possibly) transfinite reductions. For example, $\mathsf{nats}(0) \twoheadrightarrow 0 : \mathsf{s}\,0 : \mathsf{s}\,\mathsf{s}\,0 : \dots$.

For example, we can implement scalar multiplication of streams as

$$\mathsf{sca}(n, m : x) \to (n \times m) : \mathsf{sca}(n, x)$$

with $\times$ is given as in the previous section. One can easily observe that $\mathsf{sca}(\mathsf{s}\,\mathsf{s}\,0, \mathsf{nats}(0))$ enumerates the even numbers.

Obviously, each of the defined symbols $+$, $\times$, and sca in this system certainly specifies the intended function, and the system is thus productive. However, infinitary normalization of the system fails; there *are* some well-sorted and ground terms having no infinitary normal form (consider $0 \times \mathtt{sss}\ldots$ for example). This failure of infinitary normalization is caused by the infinite term $\mathtt{sss}\ldots$, which does not represent any natural number. We regard such terms as $\mathtt{sss}\ldots$ as not well-sorted from the semantical viewpoint, and want to exclude them. Thereby, the notion of *proper terms* arises as those we are concerned with.

Concept 4 A term is *proper* if it is well-sorted from the semantical viewpoint. ◻

The clause 'well-sorted from the semantical viewpoint', and hence the notion of *properness*, can be technically expressed as follows:

1. The term must be locally well-sorted. Namely, we disallow type-mismatched expressions such as $0 + (0 : 0 : 0 : \ldots)$. The term $(0 : 0 : 0 : \ldots)$ is of the sort $\mathsf{STREAM}_{\mathsf{NAT}}$, while the symbol $+$ is typed $\mathsf{NAT} \times \mathsf{NAT}$.

2. The term must not contain an infinite nesting of inductive constructors, where inductive constructors are constructor symbols used to construct inductive objects. In the example of discourse, 0 and $\mathtt{s}$ are inductive constructors; ':' is a coinductive constructor.

We call sorted infinitary term rewriting systems with the above explicit coinductive and inductive construction '*semantically sorted systems*', which will be formalized in Section 1.2.

The conclusion of this analysis is that in order to obtain the infinitary version of Concept 3 we have just to replace the word 'well-sorted' by 'proper', and the symbol $\twoheadrightarrow$ (finite reduction) by $\twoheadrightarrow\!\!\!\!\twoheadrightarrow$ (infinitary reduction).

Concept 5 A semantically sorted system is *productive* if, for every proper ground term t, there exists a unique constructor normal form s such that $t \twoheadrightarrow\!\!\!\!\twoheadrightarrow s$. ◻

The above description of productivity gives an abstract view of Concept 1, abstracted away from semantics, into the framework of term rewriting. It should be remarked that Concept 5 is not relevant to the semantical or representational perspective but only to the computational perspective.

0.3 The concept of algorithmicity

Algorithmic term rewriting systems were originally introduced in [27] as a scheme of purely functional specifications of objects employing possibly

mutually dependent induction and coinduction for their definition. In the present work we introduce the intermediate framework of semantically sorted systems, so that algorithmic systems can be recognized as semantically sorted systems that are algorithmic. There, the property of algorithmicity is to semantically sorted systems what orthogonality is to ordinary term rewriting systems.

Concept 6 An algorithmic term rewriting system should be ...

1. *Left-linear*. Employing non-left-linear reduction entails deciding syntactic identity of terms. Especially in an infinitary term rewriting system, that would cause some undesirable behaviour of transfinite reduction (e.g. failure of Compression Lemma). So, we consider only left-linear systems.

2. *Functional*, viz. each computation step is performed at the occurrence of a defined symbol with patterns given as constructor contexts.

3. *Locally deterministic*, viz. the applicable rule is unique for each position in a term.

4. *Case-exhaustive*. We are concerned with specifying total functions. ◻

With the above restrictions satisfied, the system can be perceived to specify the *algorithm* to compute each function. So, we shall call those systems 'algorithmic'.

0.4 Background

0.4.1 The subject of productivity

The word 'productive' originates from Sijtsma [44] as a natural strengthening of the weak infinitary normalization property, as the specification of a stream well-defines the whole elements of the stream. For a typical example which is infinitarily normalizing but not productive, consider the one-rule system

$$\mathsf{f}(\bullet : \bullet : x) \to \bullet : \mathsf{f}(\mathsf{f}(x))$$

This system is infinitarily normalizing, but the infinitary normal form of the term $\mathsf{f}(\bullet : \bullet : \dots)$, which is

$$\mathsf{f}(\bullet : \bullet : \dots) \to \bullet : \mathsf{f}(\bullet : \mathsf{f}(\bullet : \mathsf{f}(\bullet : \dots)))$$

contains defined function symbols f. So, the stream cannot be 'fully evaluated'. The system is thus not productive.

There is a large arsenal of investigations on productivity of stream definitions, including [7, 14, 25, 37, 49]. Roughly speaking, this approach

attempts to estimate how much 'guarded [45]' the recursive definition of the stream is. In other words, in this approach we are indifferent to the identity of each component of the stream but only concerned with the quantity of the evaluated part of the stream. This quantitative analysis is formalized in [17], and based on it, [16] gives the optimal analysis among the quantitative analyses. On the other hand, productivity of stream specifications is closely related to functional programming, and also to higher-order proof-assistants. Since the structure of streams employs recursion nested in corecursion, it requires a careful treatment to deal with streams in functional programming with lazy evaluation or proof-assistants such as Coq [9]. From this viewpoint, the notion of productivity has been discussed also in [1, 4, 22, 38].

In the present work, we extend the framework to allow infinite structure also in data. So, we deal with it in the framework of algorithmic term rewriting systems as discussed in the preceding sections. We wish to invent a feasible, sophisticated theory to ensure productivity of algorithmic systems, such as iterative path order* for termination of finitary term rewriting systems.

0.4.2 The subject of tree ordinals

As to a paradigmatic example where mutually nested induction and coinduction appear, we think of *tree ordinals*.† Tree ordinals are devised to give a simple notation for countable ordinals. The set $\mathbf{\Omega}$ of countable ordinals can be defined as the smallest set satisfying:

1. $0 \in \mathbf{\Omega}$.
2. If $\alpha \in \mathbf{\Omega}$, then $\alpha + 1 \in \mathbf{\Omega}$.
3. If A is a countable subset of $\mathbf{\Omega}$, then $\sup A \in \mathbf{\Omega}$.

So, we define expressions of tree ordinals ORD as

$$\mathsf{ORD} := \mathsf{O} \mid \mathsf{S}(\mathsf{ORD}) \mid \mathsf{ORD} : \mathsf{ORD} : \mathsf{ORD} : \dots \qquad (*)$$

where O represents 0, S represents the immediate successor, and the last one represents the supremum of the countable set. Tree ordinals can be thus regarded as a subclass of Conway numbers [8] where the lefthand-side is empty and the righthand-side is countable.

We wish to remove the infinitary context in the above definition of ORD. But, if we dealt with it as naively as

$$\mathsf{ORD} := \mathsf{O} \mid \mathsf{S}(\mathsf{ORD}) \mid \mathsf{ORD} : \mathsf{ORD}$$

*Iterative path order [34] is a method to ensure the termination property of a finitary TRS, which is closely related to recursive path order [11].

†The word 'tree ordinal' is from [10]. Dershowitz and Reingold introduce another representation with *finite* trees [13, 12].

it would cause meaningless terms such as $\mathsf{O} : \mathsf{O}$. We need ':' only to construct streams. So, we introduce another sort $\mathsf{STREAM}_{\mathsf{ORD}}$ for streams of ordinal representations:

$$\mathsf{STREAM}_{\mathsf{ORD}} :\overset{\mathbf{C}}{=} \mathsf{ORD} : \mathsf{STREAM}_{\mathsf{ORD}}$$

Then, in the previous construction ($*$), we notice the implicit transformation from $\mathsf{STREAM}_{\mathsf{ORD}}$ to ORD. In order to deal with it in the framework of sorted systems, we exhibit this transformation with the constructor L (for Limit). Thus, we reach the following construction of tree ordinals:

$$\begin{aligned} \mathsf{ORD} &:\overset{\mathbf{I}}{=} \mathsf{O} \mid \mathsf{S}(\mathsf{ORD}) \mid \mathsf{L}(\mathsf{STREAM}_{\mathsf{ORD}}) \\ \mathsf{STREAM}_{\mathsf{ORD}} &:\overset{\mathbf{C}}{=} \mathsf{ORD} : \mathsf{STREAM}_{\mathsf{ORD}} \end{aligned}$$

Thereby, mutual recursion between inductive and coinductive construction arises.

As a standard reference of ordinals we refer to [42]. Section 6 of [21] is comprehensive and enough as a preparation for reading Chapter 4 of the thesis. Further information is given in [43].

0.5 Contribution and overview

Chapter 0 is this chapter. In the remainder of the chapter we give a chapter overview of the thesis, simultaneously pointing out the contribution of the present work. The next section will present some notations used throughout the thesis.

In Chapter 1, we introduce the framework of semantically sorted infinitary term rewriting systems, and state the properties of algorithmicity and productivity. We mention also other properties of semantically sorted systems. The notions of semantically sorted systems and algorithmic systems are introduced to fill the gap between ordinary infinitary term rewriting and functional programming with lazy evaluation.

Chapter 2 provides some technical matters, which will be mainly used in Chapter 3.

In Chapter 3, we investigate a criterion for productivity of the algorithmic system. One of the ultimate goals of this work is to detect productivity or nonproductivity of algorithmic systems by an automatic or systemic procedure. In other words, an algorithm which takes an algorithmic system and gives as output productive/nonproductive/unknown. Since productivity of the algorithmic system is generally undecidable[‡], it is unavoidable to see 'unknown' for some systems (of course, we wish the algorithm to answer 'unknown' for as few systems as possible). The criterion we present

[‡]Productivity of stream specifications is already undecidable [17].

text item		end-marker
Definition		◪
Theorem, **Lemma**, **Corollary**, **Proposition**	followed by the proof	unmarked
	with the proof omitted	□
	with the proof postponed	⊟
Proof		□
Remark		⧈
Example		⊡
Concept		⊙

Table 3: *End-markers for text items*

gives the important foundation for implementing such an algorithm, and the chapter is thus the main contribution of the thesis. With that criterion, productivity of the algorithmic system can be proved by finding some functions on countable ordinals satisfying certain inequalities. In that manner, it helps us to prove productivity of some systems by hand, as will be demonstrated in Chapter 4. However, in general, analyzing inequalities between ordinal functions is a complex matter. Automatic detection of productive systems is one of our ongoing future work.

Chapter 4 studies arithmetic on tree ordinals, providing examples of productive algorithmic systems. We prove productivity of each system using the criterion presented in Chapter 3.

0.6 Notations

We mark the end of each text item (Definition, Theorem, etc.) as shown in Table 3. We have already used the marker for Concept.

The set of natural numbers is denoted by $\mathbf{N}$. The set $\mathbf{N}_+ = \mathbf{N} \setminus \{0\}$ consists of positive integers. The set of countable ordinals is denoted by $\mathbf{\Omega}$. The class of all the ordinals is denoted by $\mathbf{On}$. We follow the von Neumann-Bernays-Gödel style [3]; each ordinal is regarded as the set containing all the preceding ordinals.

Given a set A, we write A^∞ to denote the set of possibly infinite sequences of elements of A. The set of finite and infinite sequences is respectively A^* and A^ω. Infinite sequences will also be called *streams*. Clearly we have $A^\infty = A^* \uplus A^\omega$, where $\uplus$ denotes disjoint union.

The empty sequence is denoted by ϵ. Given $p \in A^*$ and $q \in A^\infty$, their concatenation is denoted by $p \cdot q$. Given $p \in A^*$ and $q \in A^\infty$, if there is $r \in A^\infty$ such that $p \cdot r = q$, then p is a *prefix* of q, and we write $p \trianglelefteq q$. In this case we write q/p to denote that r. Note that q/p is uniquely determined whenever $p \trianglelefteq q$. If moreover $p \neq q$, p is a *proper prefix* of q, and we write

$p \lhd q$. We say that two positions p and q are *parallel* if neither $p \trianglelefteq q$ nor $q \trianglelefteq p$.

The finite sequences of positive integers $\mathbf{N}_+^*$ will play an important role to describe positions in terms. We will thus often refer to an element of $\mathbf{N}_+^*$ as a *position*. A positive integer itself is often regarded as a position of length 1. We use '$\times$' instead of '$\cdot$' to denote multiplication on $\mathbf{N}$ to avoid confusion with concatenation. As a position, the length of $p \in \mathbf{N}_+^*$ is called *depth* of p, and written $|p|$.

Given a partial function f, we write $\mathrm{dom}(f)$ and $\mathrm{img}(f)$ to denote the *domain* and the *image* (or range) of f, respectively:

$$\mathrm{dom}(f) = \{a \mid f(a) \text{ is defined}\}$$
$$\mathrm{img}(f) = \{f(a) \mid a \in \mathrm{dom}(f)\}$$

The composition of the relations is denoted by '$\cdot$'. For example, we write $x \to \cdot \searrow z$ if there exist some y such that $x \to y$ and $y \searrow z$.

We write to $(-)_n$ to refer to the n-th element of a vector. For example, we have

$$((0,1,2) + (3,4,5))_1 = 3$$
$$((0,1,2) + (3,4,5))_2 = 5$$
$$((0,1,2) + (3,4,5))_3 = 7$$

We often use $\wedge$ and $\vee$ to denote minimum and maximum, respectively;

$$x \wedge y = \min\{x, y\}$$
$$x \vee y = \max\{x, y\}$$

Chapter 1

Algorithmic term rewriting systems

In this chapter, we formalize a scheme for function specifications employing both inductive and coinductive constructions, which we shall call a *semantically sorted system*. The similar concept is explored in [23] from an algebraic view.

An algorithmic system is thought of as a semantically sorted system with some requirements. Figure 4 shows an intuitive sketch. In Section 1.1 we introduce infinitary rewriting systems following [29] with minor changes for the sorted framework. And then, in Section 1.2 we introduce inductive/coinductive sort discipline and the notion of constructors to formalize semantically sorted systems. With that inductive/coinductive discipline of sorts, the notion of *proper terms*, as 'semantically meaningful'

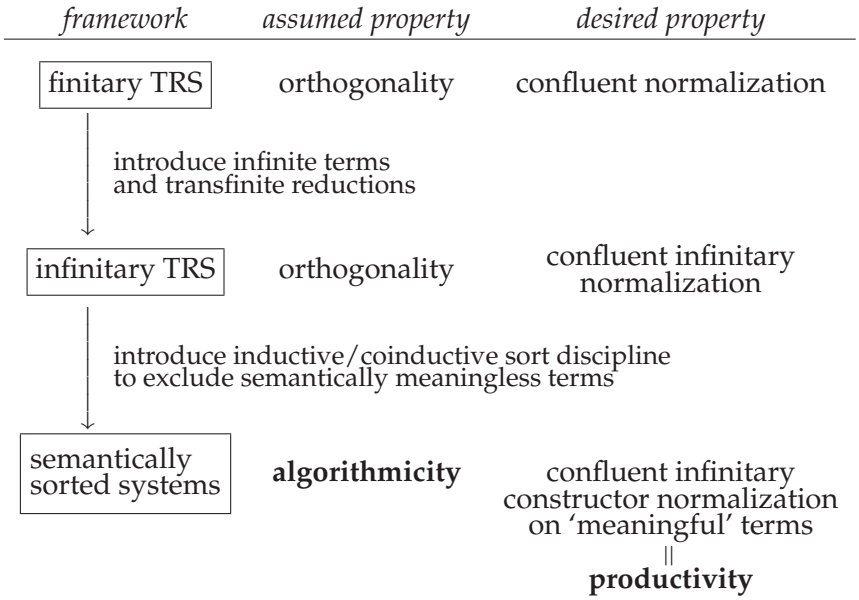


Figure 4: *Semantically sorted systems, algorithmicity, and productivity*

terms, arises. Then we state some properties of semantically sorted systems, including productivity. Section 1.3 defines the algorithmicity property of semantically sorted systems. Algorithmicity of the system is a property as natural as orthogonality of the TRS. Since algorithmicity is a natural strengthening of orthogonality, many useful properties result from the orthogonality of the algorithmic system. Section 1.4 recalls some of those properties.

1.1 Infinitary term rewriting systems

First of all, we formulate (possibly infinite) terms with explicit sorting.

1.1.1 Sorts, symbols, and terms

Definition 7 (terms) The set of *terms* is determined by the tuple $\langle \mathcal{S}, \Sigma, \mathcal{X} \rangle$ where

1. The set $\mathcal{S}$ consists of possibly infinitely many *sorts*.
2. The set Σ consists of finitely many *function symbols*, each is equipped with a first-order type $S_1 \times \cdots \times S_n \to T$, where $S_1, \ldots, S_n, T \in \mathcal{S}$. When $f \in \Sigma$ has a type $S_1 \times \cdots \times S_n \to T$, we write $\mathrm{ar}(f)$, $\mathrm{in}(f, i)$, and $\mathrm{out}(f)$ to denote n, S_i, and T, respectively, for $i = 1, \ldots n$.
3. The set $\mathcal{X}$ consists of infinitely many *variable symbols*, each is sorted by $\mathcal{S}$. We assume that for each sort $S \in \mathcal{S}$ there exist infinitely many variable symbols of the sort S. We write $\mathrm{out}(x)$ to denote the sort of a variable symbol x. For convenience, we set $\mathrm{ar}(x) = 0$ for every $x \in \mathcal{X}$.

We assume $\Sigma \cap \mathcal{X} = \varnothing$.

Given $\mathcal{S}$, Σ, and $\mathcal{X}$, a term is a partial function $t : \mathbf{N}_+^* \rightharpoonup (\Sigma \cup \mathcal{X})$ satisfying the following conditions.

1. The empty sequence ϵ is in the domain. The position ϵ or the symbol $t(\epsilon)$ is called the *root* of the term.
2. For every $p \in \mathrm{dom}(t)$ and every $n \in \mathbf{N}_+$, the position $p \cdot n$ is in the domain of t if and only if $n \leqslant \mathrm{ar}(t(p))$.
3. If $p \cdot n \in \mathrm{dom}(t)$, then $\mathrm{in}(t(p), n) = \mathrm{out}(t(p \cdot n))$.

Roughly speaking, the first condition guarantees non-emptiness of a term. The second one ensures that the number of branches at each position is consistent with the arity of the symbol by which the position is labeled. And the last one describes local well-sortedness. The *sort of a term* t is given as $\mathrm{srt}(t) = \mathrm{out}(t(\epsilon))$.

We write $\mathcal{T}$ to denote the set of terms. Syntactic identity on terms is denoted by $\equiv$. Formally, $t \equiv s$ iff $\mathrm{dom}(t) = \mathrm{dom}(s)$ and $t(p) = s(p)$ for all applicable p.

A term is *finite* if its domain is finite. The set of finite terms is denoted by $\mathcal{F}$. A term is *ground*, or *closed*, if it contains no variable symbol, i.e. t is ground if $\mathrm{img}(t) \cap \mathcal{X} = \varnothing$. The set of ground terms is denoted by $\mathcal{G}$. ◿

Proposition 8 *For every term t, the domain of t is countable.*

Proof: The domain is a subset of $\mathbf{N}_+^*$, which is countable. □

1.1.2 Operations on terms

We provide some notation devices on terms.

Definition 9 (subterms) Given a term t and a position $p \in \mathrm{dom}(t)$, the *subterm* t/p is constructed as

$$(t/p)(q) = \begin{cases} t(p \cdot q) & (\text{if } p \cdot q \in \mathrm{dom}(t)) \\ \text{undefined} & (\text{otherwise}) \end{cases}$$ ◿

Definition 10 (substitution) Given a term t and a function $\sigma : \mathcal{X} \to \mathcal{T}$ satisfying $\mathrm{out}(\sigma(x)(\epsilon)) = \mathrm{out}(x)$ for all $x \in \mathcal{X}$, the term $t\sigma$ is constructed as

$$(t\sigma)(p) = \begin{cases} \sigma(t(q))(p/q) & \left(\text{if } \exists q \in \mathrm{dom}(t), \begin{array}{l} t(q) \in \mathcal{X},\ q \trianglelefteq p, \\ p/q \in \mathrm{dom}(\sigma(t(q))) \end{array}\right) \\ t(p) & (\text{if } p \in \mathrm{dom}(t),\ t(p) \notin \mathcal{X}) \\ \text{undefined} & (\text{otherwise}) \end{cases}$$

which formalizes the ordinary substitution.

We often write $\{x_1 \mapsto t_1, \ldots, x_n \mapsto t_n\}$ to denote the function $\sigma : \mathcal{X} \to \mathcal{T}$ defined by

$$\sigma(x) \equiv \begin{cases} t_i & (\text{if } \exists i.\ x = x_i) \\ x & (\text{otherwise}) \end{cases}$$

◿

Definition 11 (replacement) Given terms t, s, and a position $p \in \mathrm{dom}(t)$ satisfying $\mathrm{out}(s(\epsilon)) = \mathrm{out}(t(p))$, the term $t\{p \mapsto s\}$ is the result of replacing the subterm of t at the position p by s, i.e.

$$t\{p \mapsto s\}(q) = \begin{cases} t(q) & (\text{if } q \in \mathrm{dom}(t),\ p \not\trianglelefteq q) \\ s(q/p) & (\text{if } p \trianglelefteq q,\ q/p \in \mathrm{dom}(s)) \\ \text{undefined} & (\text{otherwise}) \end{cases}$$

◿

1.1.3 Rules and reductions

Definition 12 (rules) A *rewrite rule* is a pair $(l, r) \in \mathcal{F} \times \mathcal{F}$ satisfying the following conditions:

1. The lefthand-side l is not a variable, i.e. $l(\epsilon) \notin \mathcal{X}$.
2. Variables occurring in the righthand-side occurs also in the lefthand-side, i.e. $\mathrm{img}(l) \cap \mathcal{X} \supseteq \mathrm{img}(r) \cap \mathcal{X}$.
3. The rule is sort-consistent, i.e. $\mathrm{srt}(l) = \mathrm{srt}(r)$.

We will often refer to a rewrite rule just as a *rule*. ◿

Definition 13 (single-step reduction) Let $t \in \mathcal{T}$, $p \in \mathrm{dom}(t)$, $(l, r) \in \mathcal{R}$, and $\sigma : \mathcal{X} \to \mathcal{T}$ such that $t/p \equiv l\sigma$. Then, we write $t \to_p s$, where $s \equiv t\{p \mapsto r\sigma\}$. This p is called the position of the contracted redex. We often omit the subscript p to write $t \to s$ when the position of contracted redex is not important. ◿

Directly from the definition, a lemma follows:

Lemma 14 *Let* $t \to_p s$. *Then:*

- *For every* $q \in \mathrm{dom}(t)$ *such that* $q \leqslant p$, *we have* $t/q \to_{p/q} s/q$.
- *For every* $q \in \mathrm{dom}(t)$ *which is parallel to* p, *we have* $t/q \equiv s/q$. □

A term t is called a *normal form* if there exists no term s such that $t \to s$. The set of normal forms is denoted by $\mathcal{N}$.

We formalize transfinite reduction following [30].

Definition 15 (transfinite reduction) Let $\langle t_\iota \rangle_{\iota \leqslant \alpha}$ be a transfinite sequence of terms of length $\alpha \in \mathbf{\Omega}$. Then, $\langle t_\iota \rangle$ is *convergent* if for every limit ordinal $\lambda \leqslant \alpha$ and every $p \in \mathrm{dom}(t_\lambda)$, there exists some $\beta < \lambda$ such that, for every ι, $\beta \leqslant \iota < \lambda$ implies $t_\iota(p) = t_\lambda(p)$.

A convergent sequence $\langle t_\iota \rangle \leqslant \alpha$ is a *convergent reduction sequence* if the following conditions are satisfied:

1. For every $\iota < \alpha$, we have $t_\iota \to t_{\iota+1}$.
2. For every limit ordinal $\lambda \leqslant \alpha$ and every $n \in \mathbf{N}$, there exists some $\beta < \lambda$ such that $\beta \leqslant \iota < \lambda$ implies $|p_\iota| > n$, where p_ι is the position of the contracted redex at the single-step reduction $t_\iota \to t_{\iota+1}$. This condition is known as *strong convergence*.

In this case, we write $t_0 \to^{\alpha} t_\alpha$. When the length of reduction is not important, we write also $t_0 \twoheadrightarrow t_\alpha$. Note that $\to^1$ is identical to the single step reduction $\to$. If $\alpha < \omega$, we write $t_0 \twoheadrightarrow t_\alpha$.

An infinite reduction sequence $t_0 \to t_1 \to t_2 \to t_3 \to \dots$ is called *divergent* if there exists $p \in \mathbf{N}_+^*$ such that $t_i \to_p t_{i+1}$ for infinitely many $i \in \mathbf{N}$. ◿

Definition 16 (stability of reduction) A transfinite reduction $t \twoheadrightarrow s$ is *strongly* p*-stable* if for any single-step reduction in $t \twoheadrightarrow s$, the position of the contracted redex is not a prefix of p. A transfinite reduction $t \twoheadrightarrow s$ is *weakly* p*-stable* if it is strongly q-stable for every $q \lhd p$. Note that a reduction is always weakly ϵ-stable. We use the subscript $\not\lhd p$ or $\not\trianglelefteq p$ to denote weak or strong p-stability, respectively. Especially, we use the subscript $\neg\epsilon$ to denote strong ϵ-stability, which is called root-stability in the literature. ◿

1.1.4 Constructors

Given a set of rules, we formalize the division of function symbols into constructor function symbols and defined function symbols as mentioned in Concept 2.

Definition 17 A *defined function symbol* (*defined symbol* for short) is a function symbol which occurs at the root of the lefthand-side of a rule. A *constructor function symbol* (*constructor* for short) is a function symbol which never occurs at the root of the lefthand-side of any rule. The sets of defined symbols and constructors are denoted by $\mathcal{C}$ and $\mathcal{D}$, respectively. Formally,

$$\mathcal{D} = \{l(\epsilon) \mid (l, r) \in \mathcal{R}\}$$
$$\mathcal{C} = \Sigma \setminus \mathcal{D}$$

Clearly, every function symbol is either a defined symbol or a constructor. ◿

A term t is a *constructor normal form* if it contains only constructors, i.e. $\mathrm{img}(t) \subseteq \mathcal{C}$. The set of constructor normal forms is denoted by $\mathcal{V}$.

Obviously, every constructor normal form is ground and forms a normal form. We say that a term t *has* a constructor normal form s if $t \twoheadrightarrow s$ and $s \in \mathcal{V}$. Moreover, if this s is unique, we write $\mathrm{cnf}(t)$ to denote that s.

1.1.5 Sorted systems

To close the section, we describe the formalization of the framework of sorted infinitary term rewriting systems.

Definition 18 A sorted infinitary term rewriting system is determined by a tuple $\langle \mathcal{S}, \Sigma, \mathcal{X}, \mathcal{R} \rangle$, where each symbol is equipped with a sort specification ar, in, and out. ◿

Example 19 In the above formalization, the system of addition and multiplication on natural numbers

$$\begin{aligned} n + 0 &\to n \\ n + s\,m &\to s(n + m) \\ n \times 0 &\to 0 \\ n \times s\,m &\to (n \times m) + n \end{aligned}$$

as presented in the Introduction is given as

$$\begin{aligned} +(n, 0()) &\to n \\ +(n, s(m)) &\to s(+(n, m)) \\ \times(n, 0()) &\to 0() \\ \times(n, s(m)) &\to +(\times(n, m), n) \end{aligned}$$

where $\mathcal{S} = \{\mathsf{NAT}\}$, $\mathcal{C} = \{0, s\}$, $\mathcal{D} = \{+, \times\}$, and $\mathcal{X} = \{n, m\}$. For the sake of readability, we will often omit parentheses after a nullary or unary function symbol, and use infix notation for some binary function symbols. ⊡

1.2 Semantically sorted systems

As seen in the Introduction, an ordinary sorted infinitary term rewriting system possibly contains some meaningless infinite terms such as $s\,s\,s\,s\,s\ldots$ in the system of streams of natural numbers. We wish to exclude those terms. The next thing to be dealt with is to formalize this exclusion.

1.2.1 Inductive and coinductive sorts

Definition 20 Let $\mathcal{S} = \mathcal{S}^{\mathbf{I}} \uplus \mathcal{S}^{\mathbf{C}}$. The set $\mathcal{S}^{\mathbf{I}}$ consists of *inductive* sorts, and the set $\mathcal{S}^{\mathbf{C}}$ consists of *coinductive* sorts. This division of $\mathcal{S}$ yields the characteristic functions $\chi^{\mathbf{I}}, \chi^{\mathbf{C}} : \mathcal{S} \to \{0, 1\}$ given by

$$\chi^{\mathbf{I}}(S) = \begin{cases} 1 & (\text{if } S \in \mathcal{S}^{\mathbf{I}}) \\ 0 & (\text{if } S \in \mathcal{S}^{\mathbf{C}}) \end{cases} \qquad \text{and} \qquad \chi^{\mathbf{C}}(S) = \begin{cases} 0 & (\text{if } S \in \mathcal{S}^{\mathbf{I}}) \\ 1 & (\text{if } S \in \mathcal{S}^{\mathbf{C}}) \end{cases}$$

We write $\chi^{\mathbf{I}}_{\text{in}}(f, n)$, $\chi^{\mathbf{C}}_{\text{in}}(f, n)$, $\chi^{\mathbf{I}}_{\text{out}}(f)$, and $\chi^{\mathbf{C}}_{\text{out}}(f)$ to denote $\chi^{\mathbf{I}}(\text{in}(f, n))$, $\chi^{\mathbf{C}}(\text{in}(f, n))$, $\chi^{\mathbf{I}}(\text{out}(f))$, and $\chi^{\mathbf{C}}(\text{out}(f))$, respectively. ◿

Example 21 The system of streams of natural numbers is constructed by $\Sigma = \{0, s, \mathsf{cons}^{\mathsf{NAT}}\}$ with

$$\begin{aligned} 0 &: && () \to \mathsf{NAT} \\ s &: && \mathsf{NAT} \to \mathsf{NAT} \\ \mathsf{cons}^{\mathsf{NAT}} &: && \mathsf{NAT} \times \mathsf{STREAM}_{\mathsf{NAT}} \to \mathsf{STREAM}_{\mathsf{NAT}} \end{aligned}$$

where $n : x$ is regarded as an abbreviated form of $\mathsf{cons}^{\mathsf{NAT}}(n, x)$. Here, we stipulate $\mathcal{S}^{\mathbf{I}} = \{\mathsf{NAT}\}$ and $\mathcal{S}^{\mathbf{C}} = \{\mathsf{STREAM}_{\mathsf{NAT}}\}$. ⊡

1.2.2 Semantically sorted term rewriting systems

With the above division of constructors, we formalize the framework of sorted infinitary term rewriting systems which can deal with the objects employing inductive and coinductive constructions. We shall call such systems 'semantically sorted systems'.

Definition 22 A *semantically sorted (infinitary term rewriting) system* is determined by a tuple $\langle \mathcal{S}^{\mathbf{I}}, \mathcal{S}^{\mathbf{C}}, \Sigma, \mathcal{X}, \mathcal{R} \rangle$, where each function symbol is equipped with a sort specification ar, in, and out. ◪

Since every semantically sorted term rewriting system can be regarded as a sorted infinitary term rewriting system, we already have the notions of terms ($\mathcal{T}$), ground terms ($\mathcal{G}$), finite terms ($\mathcal{F}$), normal forms ($\mathcal{N}$), and constructor normal forms ($\mathcal{V}$).

Now that the sorts have been divided into inductive and coinductive sorts, we exclude meaningless terms such as $\mathsf{sssss}\ldots$ as mentioned in the Introduction. We also disallow infinite nesting of defined function symbols. In order to do that, we introduce the notion of 'path' and 'vicious path' in a term.

For an intuitive sketch, see Figure 5. We divide the space of function symbols by two ways: defined symbol or constructor ($\Sigma = \mathcal{D} \uplus \mathcal{C}$); inductive or coinductive output sort ($\mathcal{S} = \mathcal{S}^{\mathbf{I}} \uplus \mathcal{S}^{\mathbf{C}}$). We exclude infinite nesting of grayed symbols there.

Definition 23 Let t be a term. Then, an *infinite path* in t is an infinite sequence $p \in \mathbf{N}_+^\omega$ such that $\{p' \mid p' \lhd p\} \subseteq \mathrm{dom}(t)$. We refer to an infinite path just as a *path*, since a finite path can be specified by a position.

A path p in a term t is *vicious* if there exist infinitely many $p' \lhd p$ such that $t(p') \in \mathcal{D}$ or $\mathrm{out}(t(p')) \in \mathcal{S}^{\mathbf{I}}$. ◪

Roughly speaking, a term is meaningless if the term contains a vicious path. A vicious path is a *constructor vicious path* if the path contains no defined function symbol; a *functional vicious path* if the path contains

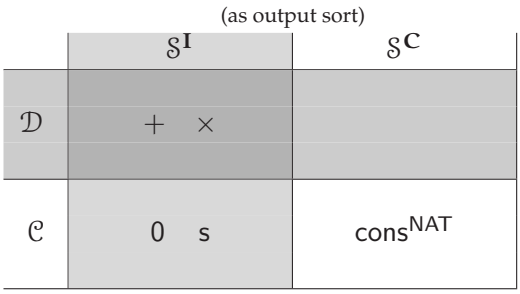


Figure 5: *Four kinds of function symbols*

infinitely many defined function symbols. Note that a vicious path is neither a constructor vicious path nor a functional vicious path when the path contains finitely many defined function symbols.

Definition 24 (proper terms) A term is *proper* if the term contains no vicious path. The set of proper terms is denoted by $\mathcal{T}^\star$. ◪

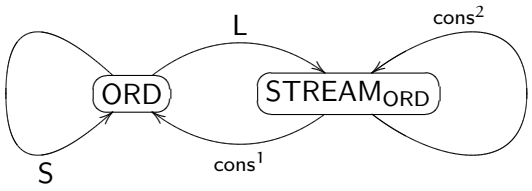
We use the superscript $\star$ to indicate properness; we write $\mathcal{G}^\star$ and $\mathcal{V}^\star$ to denote $\mathcal{T}^\star \cap \mathcal{G}$ and $\mathcal{T}^\star \cap \mathcal{V}$, respectively.

Remark 25 Provided that $\mathcal{S}$ is finite, vicious paths in ground terms can be characterized by an ω-tree automaton [39, 47], where the states consist of the sorts and each constructor symbol $f : S_1 \times \cdots \times S_n \to T$ induces the transitions $T \xrightarrow{f^i} S_i$. Each path is recognized as a run of the automaton, and vicious paths are those containing infinitely many steps through the state of an inductive sort.

For example, the system of tree ordinals, as in the Introduction,

$$\begin{aligned} \mathsf{ORD} &:\overset{\mathbf{I}}{=} \mathsf{O} \mid \mathsf{S}(\mathsf{ORD}) \mid \mathsf{L}(\mathsf{STREAM}_{\mathsf{ORD}}) \\ \mathsf{STREAM}_{\mathsf{ORD}} &:\overset{\mathbf{C}}{=} \mathsf{ORD} : \mathsf{STREAM}_{\mathsf{ORD}} \end{aligned}$$

induces the following automaton:



Runs of vicious paths are those going through the state ORD infinitely many times. ⧈

As described in Concept 5, we are concerned with infinitary computation of proper ground terms. Therefore, the terms to be considered are those we can reach from proper terms with transfinite reduction. It should be noticed that every finite term is proper.

Definition 26 (initially proper terms) A term t is *initially proper* if there exists a proper term s such that $s \twoheadrightarrow t$. The set of initially proper terms is denoted by $\mathcal{T}^{\star\twoheadrightarrow}$. Similarly, a term t is *initially proper ground* if there exists a proper ground term s such that $s \twoheadrightarrow t$. The set of initially proper ground term is denoted by $\mathcal{G}^{\star\twoheadrightarrow}$. Since the property of being a ground term is preserved under transfinite reduction, we have $\mathcal{G}^{\star\twoheadrightarrow} = \mathcal{T}^{\star\twoheadrightarrow} \cap \mathcal{G}$. ◿

Lemma 27 *If a term* t *is initially proper or initially proper ground, then so is each subterm of* t*.*

Proof: Let t be an initially proper term, and $p \in \mathrm{dom}(t)$. Then, there exists $s \in \mathcal{T}^\star$ such that $s \twoheadrightarrow t$. By strong convergence of the reduction, we can find s' such that $s \twoheadrightarrow s' \twoheadrightarrow_{\not\trianglelefteq p} t$. Thus, $s'/p \twoheadrightarrow t/p$, implying $t/p \in \mathcal{T}^{\star\twoheadrightarrow}$. The claim can be similarly proved for initially proper ground terms. □

The subsets of terms given so far and set inclusion relations among them are depicted in Figure 6, where the origin of an arrow is a subset of the target of the arrow.

Next, we define some properties of semantically sorted term rewriting systems. We will overload some words and acronyms in ordinary rewritings such as WN. In particular, we use these notions only for proper ground terms. So, we use e.g. WN for what would usually be called 'ground-WN', restricted to proper terms.

Definition 28

1. A system is *productive* (PR) if every proper ground term has a unique proper constructor normal form, i.e. for every $t \in \mathcal{G}^\star$, there uniquely exists $s \in \mathcal{V}^\star$ such that $t \twoheadrightarrow s$.

2. A system is *strongly productive* (SPR) if the system is productive and properness of terms is preserved under transfinite reduction.

3. A system is *finite-productive* if every finite ground term has a unique proper constructor normal form, i.e. for every $t \in \mathcal{F} \cap \mathcal{G}$, there uniquely exists $s \in \mathcal{V}^\star$ such that $t \twoheadrightarrow s$. ◿

Remark 29 In the above definition, we have stated the three kinds of productivity; productive, strongly productive, and finite-productive. There are obvious implications among them, i.e. strong productivity implies productivity, and productivity implies finite-productivity.

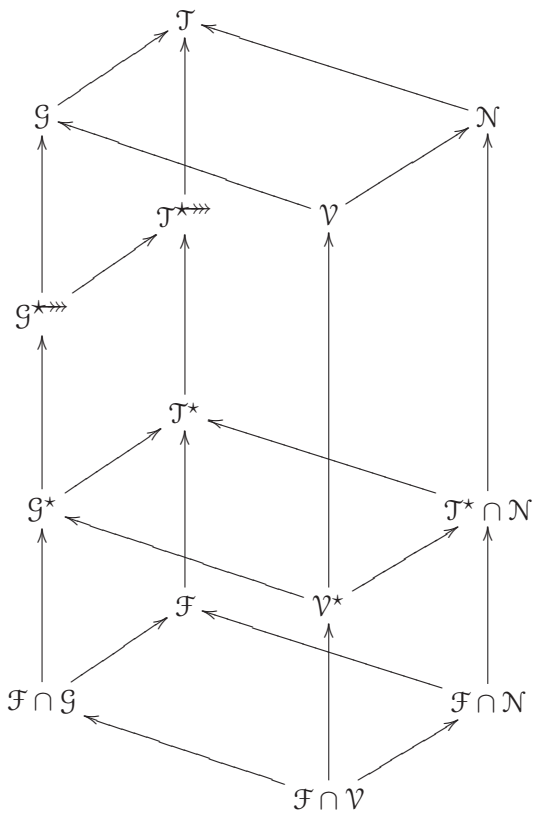


Figure 6: *Set inclusions (embeddings)*

The gap between productivity and strong productivity is in some sense similar to the gap between strong and weak normalization in finitary rewriting systems. For example, consider the following system:

$$\mathsf{K}(n, m) \to n$$
$$\mathsf{alt\_id}(n) \to \mathsf{K}(n, \mathsf{alt\_id}(n))$$

Observe that the term $\mathsf{alt\_id}(0)$ produces a functional vicious path by the reduction $\mathsf{alt\_id}(0) \twoheadrightarrow \mathsf{K}(0, \mathsf{K}(0, \mathsf{K}(0, \dots)))$, but that vicious path vanishes after infinitary normalization.

As to the gap between productivity and finite-productivity, consider the following system:

$$\mathsf{cnt0}(0 : x) \to \mathsf{s}(\mathsf{cnt0}(x))$$
$$\mathsf{nats}(n) \to n : \mathsf{nats}(\mathsf{s}\, n)$$

The function $\mathsf{cnt0}$, counting the number of leading 0, disturbs productivity of the system; $\mathsf{cnt0}(0 : 0 : 0 : \dots)$ has the non-proper normal form

$\mathsf{sss}\ldots$. On the other hand, the system is clearly finite-productive. Thus, from some practical viewpoint, finite-productivity might be an important notion. However, finite-productivity is theoretically complicated, since it is essentially relative to reachability of terms by transfinite reduction. Worse, finite-productivity is not preserved under union of systems; the union of the above system and the one-rule system

$$\mathsf{zeros} \to \mathsf{0} : \mathsf{zeros}$$

which is productive, yields a non-finite-productive system. The present work will not deal with finite-productivity. ⧈

Next we define some other properties as technical instruments in analyzing productivity.

Definition 30

1. A system is *weakly normalizing w.r.t. ground terms (*WN*)* if every proper ground term has a normal form, i.e. for every $t \in \mathcal{G}^\star$, there exists $s \in \mathcal{N}$ such that $t \twoheadrightarrow s$.

2. A system is *uniquely normalizing (*UN*)* if the normal form of each proper ground term is uniquely determined, i.e.

$$t \in \mathcal{G}^\star,\ s, s' \in \mathcal{N}\ t \twoheadrightarrow s,\ t \twoheadrightarrow s' \Rightarrow s \equiv s'$$

3. A system is *infinitarily confluent w.r.t. ground terms (*CR*)* if

$$(t \twoheadrightarrow s,\ t \twoheadrightarrow s',\ t \in \mathcal{G}^\star) \Rightarrow \exists u \in \mathcal{T}.\ s \twoheadrightarrow u,\ s' \twoheadrightarrow u$$

4. A system is *domain normalizing (*DN*)* if no initially proper ground term contains a constructor vicious path. That means, beginning with a proper ground term, if we generate a vicious path, then that path should contain some defined function symbols.

5. A system is *constructor normalizing (*CN*)* if no initially proper ground normal form contains a functional vicious path. That means, beginning with a proper ground term, if we reach an infinitary normal form, then there should be no infinite nesting of defined function symbols.

6. A system is *strongly constructor normalizing (*SCN*)* if no initially proper ground term contains a functional vicious path. That means, beginning with a proper ground term, if we generate a vicious path, then that path can contain only finitely many defined function symbols.

7. A system is *proper* if every infinitary reduct of a proper ground term is proper, i.e. $\mathcal{G}^{\star\twoheadrightarrow} = \mathcal{G}^\star$.

8. A system is *case-exhaustive (*CE*)* if every term of a form $f(t_1, \ldots, t_n)$, where $f \in \mathcal{D}$ and $t_1, \ldots, t_n \in \mathcal{V}$, forms a redex, viz. the rules cover all the patterns of constructor contexts. ◿

Lemma 31 *A semantically sorted system is* CR *if the system is* WN *and* UN. □

Lemma 32 *A semantically sorted system is proper if and only if the system is* DN *and* SCN.

Proof: (If) For a proof by contradiction, assume $t \in \mathcal{G}^{\star\twoheadrightarrow} \setminus \mathcal{G}^{\star}$. Let p be a vicious path in t. If p is a functional vicious path, then it contradicts SCN. So, p contains only finitely many defined function symbols. We can thus find a prefix q of p such that p/q contains no defined function symbols. Then, by Lemma 27, $t/p \in \mathcal{G}^{\star\twoheadrightarrow}$ and t/p contains a constructor vicious path p/q, contradicting DN.

(Only If) Trivial. □

Lemma 33 *If a semantically sorted system is* CE*, then the following conditions are equivalent:*

1. *The system is* CN*.*
2. *No initially proper ground normal form contains a defined symbol, i.e.* $(\mathcal{G}^{\star\twoheadrightarrow} \cap \mathcal{N}) \subseteq \mathcal{V}$*.*

Proof: (1⇒2) For a proof by contradiction, assume $s \in (\mathcal{G}^{\star\twoheadrightarrow} \cap \mathcal{N}) \setminus \mathcal{V}$. We will coinductively construct a path p_s in s, which forms a functional vicious path, contradicting CN.

Let $A = \{q \in \mathrm{dom}(s) \mid s(q) \in \mathcal{D}\}$. Since $s \notin \mathcal{V}$, we have $A \neq \varnothing$. Moreover, we have $A \neq \{\epsilon\}$; otherwise we have $s \notin \mathcal{N}$ by CE, which contradicts the supposition. Choose a $q \in A \setminus \{\epsilon\}$ and let $s' \equiv s/q$. Note that $s' \in \mathcal{N} \setminus \mathcal{V}$. Now, let $p_s = q \cdot p_{s'}$. Observe that p_s certainly forms a functional vicious path.

(2⇒1) Trivial. □

Lemma 34 *A semantically sorted system is productive if and only if the system is* WN*,* UN*,* DN*,* CN*, and* CE*.*

Proof: (If) Suppose that the system is WN, UN, DN, CN, and CE. Let t be a proper ground term. Then, from WN and UN, there exists a unique $s \in \mathcal{N}$ such that $t \twoheadrightarrow s$. From CN and CE with Lemma 33, $s \in \mathcal{V}$. So, any vicious path in s must be a constructor vicious path, which does not exist by DN. Hence, $s \in \mathcal{V}^{\star}$.

(Only If) Trivial. □

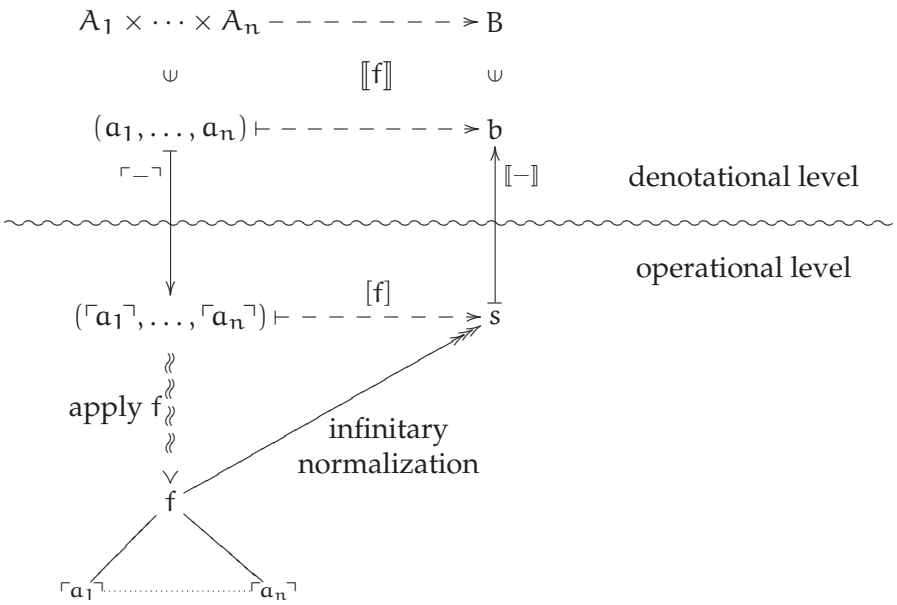


Figure 7: *Semantics of a defined function symbol*

Lemma 35 *A semantically sorted system is strongly productive if and only if the system is* WN, UN, CE *and proper.*

Proof: (If) Productivity follows from Lemmas 32 and 34. Since the system is proper, the system is strongly productive.

(Only If) Trivial. □

Remark 36 In a productive system, each defined function symbol f gives rise to a function $[f] : \mathcal{V}^\star_{in(f,1)} \times \cdots \times \mathcal{V}^\star_{in(f,ar(f))} \to \mathcal{V}^\star_{out(f)}$, given as: $[f](t_1, \ldots, t_n)$ is the normal form of $f(t_1, \ldots, t_n)$, where $\mathcal{V}^\star_S$ denotes the set of proper constructor normal forms of the sort S (see Figure 7). Section 2.6 will formalize this feature. ⧈

1.3 Algorithmicity

Following Concept 6, we define the algorithmicity property of a semantically sorted system.

Definition 37

1. A system is *non-empty* if every sort has at least one instance, i.e. for every $S \in \mathcal{S}$, there exists $t \in \mathcal{V}^\star$ such that $srt(t) = S$.
2. A rule (l, r) is *left-linear* if every variable occurs in l at most once. A system is *left-linear* if all the rules are so.

3. A rule (l, r) is *functional* if a defined symbol occurs in l only at the root, i.e. $l(p) \in \mathcal{D} \Leftrightarrow p = \epsilon$. A system is *functional* if all the rules are so.

4. A system is *locally deterministic* if there exists no root-overlapping, i.e. $(l, r), (l', r') \in \mathcal{R},\ \sigma, \sigma' : \mathcal{X} \to \mathcal{T},\ l\sigma \equiv l'\sigma' \Rightarrow l \equiv l',\ r \equiv r'$.

5. A system is *algorithmic* if the system is non-empty, left-linear, functional, locally deterministic, and case-exhaustive (see Definition 30.8). ◪

Algorithmicity of the semantically sorted system can be thought of as a natural strengthening of orthogonality of the infinitary term rewriting system. In the next section we will see that algorithmic systems are always orthogonal (Lemma 44). From orthogonality of algorithmic systems, some lemmas follow. We will deal with the proofs in the next section, as properties of orthogonal systems. Thus, as another way of the formalization of algorithmic systems, we can define those as non-empty and case-exhaustive orthogonal systems.

Lemma 38 (Compression Lemma) *If* $t \longrightarrow\!\!\!\!\!\rightarrow s$ *in an algorithmic system, then we have* $t \twoheadrightarrow s$ *or* $t \to^{\omega} s$.

Proof: Actually, left-linearity of the system already suffices to have the above implication. That will be stated as Lemma 45. □

Lemma 39 *An algorithmic system is always* UN.

Proof: Follows from Lemmas 44 and 50. □

Corollary 40 *An algorithmic system is productive if and only if the system is* WN, DN, *and* CN.

Figure 9 illustrates how the WN, DN, and CN properties contribute to productivity.

Proof: By the above lemma, the system is UN. By Definition 37, the system is CE. The claim then follows from Lemma 34. □

Corollary 41 *An algorithmic system is* SPR *if and only if the system is* WN, DN *and* SCN.

Proof: Follows similarly as the above corollary, using Lemma 35. □

Figure 8 illustrates these two corollaries.

Lemma 42 *An algorithmic system is finitarily confluent: if* $t \twoheadrightarrow s$ *and* $t \twoheadrightarrow s'$, *then there exists* $u \in \mathcal{T}$ *such that* $s \twoheadrightarrow u$ *and* $s' \twoheadrightarrow u$.

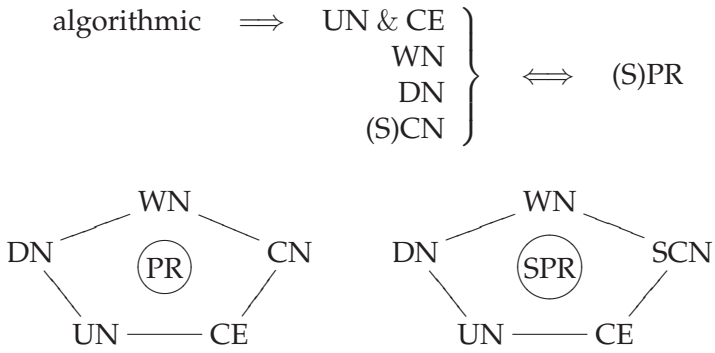


Figure 8: *Corollaries 40 and 41*

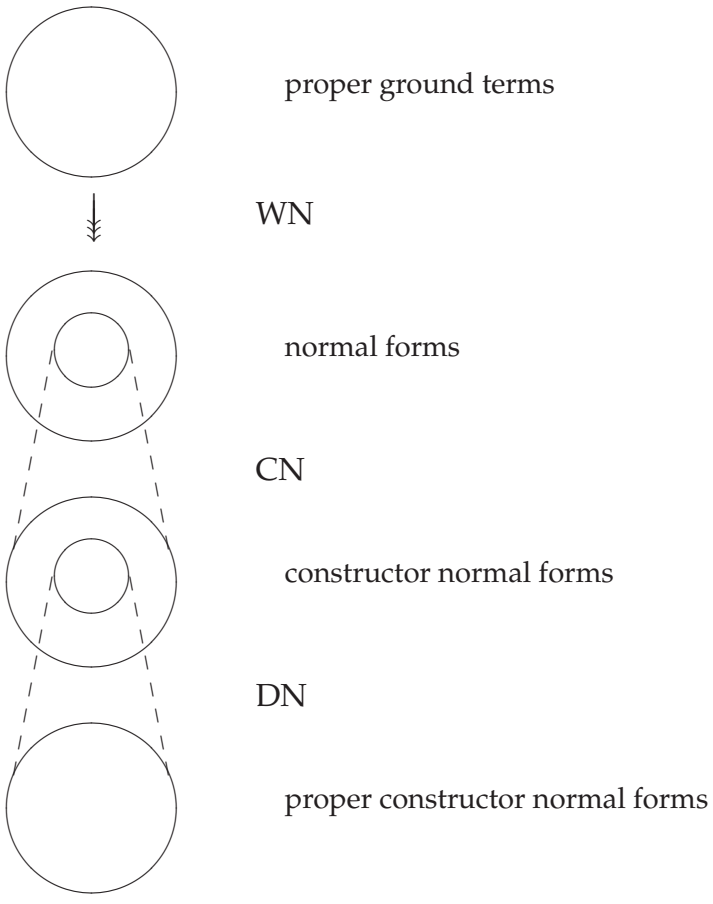


Figure 9: *Contribution of the* WN, DN, *and* CN *properties for productivity*

Proof: Follows from Lemmas 44 and 49. □

It should be remarked that productivity of the algorithmic system already suffices for having strong normalization not only of proper ground terms but also of non-ground proper terms. This fact will be stated as Corollary 53.

1.4 Orthogonality

For the last section of the chapter, we recall the orthogonality of an infinitary term rewriting system, and some properties following from orthogonality of the system. Those who accept the lemmas presented in the previous section may well skip this section.

Definition 43 Let (l, r) and (l', r') be a pair of rules. Then, they are *overlapping* if there exists a term containing redexes generated by both rules respectively, where at least one function symbol is common in the patterns. Formally, (l, r) and (l', r') are overlapping if there exist $\sigma, \tau : \mathcal{X} \to \mathcal{T}$ and $p \in \mathrm{dom}(l)$ satisfying the following conditions:

1. The term $l\sigma$, which clearly has the redex at the root with the rule (l, r), contains another redex at the position p, i.e. $(l\sigma)/p \equiv l'\tau$.
2. The redexes are not identical, i.e. $(l, r) \not\equiv (l', r')$ or $p \neq \epsilon$.
3. The redexes are not nested, i.e. $l(p)$ is not a variable symbol.

A left-linear infinitary term rewriting system is *orthogonal* if the system contains no overlapping pair of rules. ◪

Lemma 44 *An algorithmic term rewriting system is always orthogonal.*

Proof: For a proof by contradiction, assume that there exists an overlapping pair of rules (l, r) and (l', r') with $\sigma, \tau : \mathcal{X} \to \mathcal{T}$ and $p \in \mathrm{dom}(l)$ satisfying the condition as in the above definition. Then, we have $(l\sigma)/p \equiv l'\tau$. Compare the symbol at the root to have $(l\sigma)(p) = (l'\tau)(\epsilon)$. From functionality of the system, we have $l'(\epsilon) \in \mathcal{D}$ and hence $(l'\tau)(\epsilon) = l'(\epsilon) \in \mathcal{D}$. If $p \neq \epsilon$, then again from functionality we have $l(p) \notin \mathcal{D}$. Since $p \notin \mathcal{X}$ by the definition of overlapping, we have $(l\sigma)(p) = l(p) \notin \mathcal{D}$. That contradicts the assumption $(l\sigma)(p) = (l'\tau)(\epsilon)$. Therefore, p must be ϵ, implying $l\sigma = l'\tau$. Since the system is locally deterministic, we have $(l, r) \equiv (l', r')$. That contradicts the second condition of an overlapping redex. □

Lemma 45 (Compression Lemma) *If* $t \twoheadrightarrow s$ *in a left-linear infinitary term rewriting system, then there exists* $\alpha \leqslant \omega$ *such that* $t \to^{\alpha} s$. ⊟

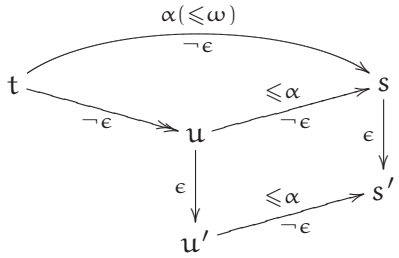


Figure 10: *Lemma 47, root-step shifting*

In order to prove the above lemma, we need two other lemmas.

Lemma 46 *If* $t \twoheadrightarrow\!\!\!\!\!\rightarrow s$, *then for every* $n \in \mathbf{N}$, *there are only finitely many single-step reductions at depth* n *in* $t \twoheadrightarrow\!\!\!\!\!\rightarrow s$.

Proof: Let $\langle t_\iota \rangle_{\iota \leqslant \alpha}$ be a convergent reduction sequence, and let

$$X = \{\iota \in \alpha \mid t_\iota \to_{p_\iota} t_{\iota+1},\ |p_\iota| = n\}$$

For a proof by contradiction, assume X is infinite. Then, there exists a limit ordinal $\lambda < \alpha$ such that $X \cap \lambda$ is also infinite. Choose the minimal λ with these properties. From the definition of convergent reduction, we can find $\beta < \lambda$ such that $\beta \leqslant \iota < \lambda \Rightarrow |p_\iota| > n$. Then, from the definition we have $X \cap \lambda \subseteq \beta$. Since $X \cap \lambda$ is infinite, β should be also infinite. Thus, β is of the form $\beta' + k$, where β' is a limit ordinal and $k \in \mathbf{N}$. Now, we have

$$X \cap \lambda = X \cap \beta = (X \cap \beta') \cup (X \cap \{\iota \mid \beta' \leqslant \iota < \beta\})$$

Since $\{\iota \mid \beta' \leqslant \iota < \beta\}$ has only finitely many elements, $X \cap \beta'$ should be infinite. That contradicts the minimality of λ. □

Lemma 47 *If* $t \to_\epsilon{}^\alpha s \to_\epsilon s'$ *and* $\alpha \leqslant \omega$ *in a left-linear system, then we can find* $u, u' \in \mathcal{T}$ *and* $\beta \leqslant \alpha$ *such that* $t \twoheadrightarrow_\epsilon u \to_\epsilon u' \to_\epsilon{}^\beta s'$ *(see also Figure 10).*

Proof: If $\alpha < \omega$, then we have $t \twoheadrightarrow_\epsilon s \to_\epsilon s' \to^0 s'$. So, letting $u \equiv s$ and $u' \equiv s'$ suffices.

Now, suppose $\alpha = \omega$. Let $(l, r) \in \mathcal{R}$ be the rule applied at the step $s \to_\epsilon s'$ with $l\sigma \equiv s$ and $r\sigma \equiv s'$. By strong convergence of the reduction $t \to_\epsilon{}^\omega s$, we can find some u such that $t \twoheadrightarrow u \to^\omega s$ where $u \to^\omega s$ is strongly p-stable for every $p \in \mathrm{dom}(l)$. Let $\langle u_\iota \rangle \leqslant \omega$ be the convergent reduction sequence with $u_0 \equiv u$ and $u_\omega \equiv s$. Then, from the definition of strong stability, we have $u_i(p) = s(p)$ for every $p \in \mathrm{dom}(l)$ and every

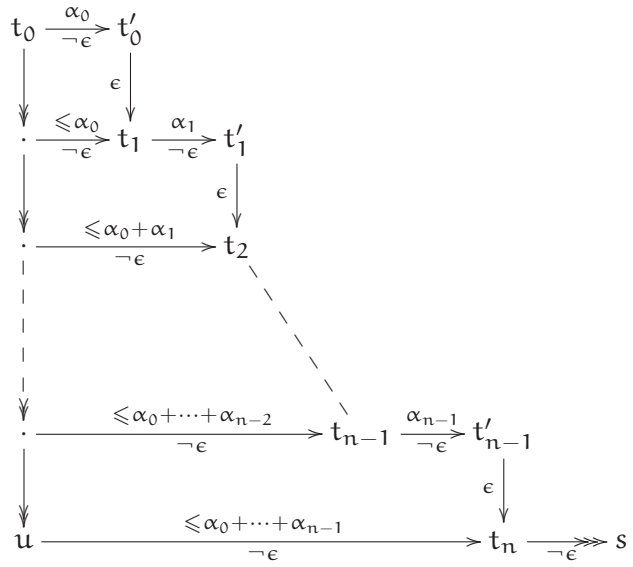


Figure 11: *Projecting root-reduction steps*

$i \in \mathbf{N}$. Thus, by left-linearity of the system, for each i we can find a $\tau_i : (\mathrm{img}(l) \cap \mathcal{X}) \to \mathcal{T}$ such that $u_i \equiv l\tau_i$. Now, let $u'_i \equiv r\tau_i$ for every i. Then, every step $u_i \to_{p_i} u_{i+1}$ is projected to $u'_i \twoheadrightarrow_{\neg\epsilon} u'_{i+1}$ as follows:

1. We can find $q \in \mathrm{dom}(l)$ such that $q \lhd p_i$ and $l(q) \in \mathcal{X}$.
2. Then, we have $u_i/q \to_{p_i/q} u_{i+1}/q$.
3. The same rule is applicable to u'_i at each of the positions $q' \cdot (p_i/q)$ such that $r(q') = l(q)$. Since $r(q') \in \mathcal{X}$, these positions are parallel.
4. Proceed with all those redexes to have $u'_i \twoheadrightarrow_{\neg\epsilon} u'_{i+1}$.

Then, the reduction $u' \equiv u'_0 \twoheadrightarrow_{\neg\epsilon} u'_1 \twoheadrightarrow_{\neg\epsilon} u'_2 \twoheadrightarrow_{\neg\epsilon} \dots$ forms a convergent reduction sequence leading to s'. □

Proof of Lemma 45: First, we define a predicate on $\mathbf{\Omega}$: a countable ordinal α is *compressible* if for every α-long transfinite reduction $t \to^\alpha s$, there exists $\beta \leqslant \omega$ such that $t \to^\beta s$. Clearly, we can rephrase the lemma as stating that every countable ordinal is compressible. We will show this by transfinite induction on $\mathbf{\Omega}$.

Suppose that every ordinal less than α is compressible. To show that α is also compressible will suffice. Suppose $t \to^\alpha s$. By Lemma 46, there are only finitely many root-reduction steps in $t \to^\alpha s$. Thus, without loss of

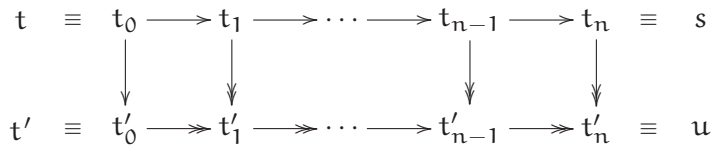


Figure 12: *Projection of a finite reduction*

generality, we can assume

$$t \equiv t_0 \to_{\backsim\epsilon}{}^{\alpha_0} t'_0 \to_\epsilon t_1 \to_{\backsim\epsilon}{}^{\alpha_1} t'_1 \to_\epsilon t_2 \to_{\backsim\epsilon}{}^{\alpha_2} t'_2 \to_\epsilon \dots \\ t_{n-1} \to_{\backsim\epsilon}{}^{\alpha_{n-1}} t'_{n-1} \to_\epsilon t_n \to_{\backsim\epsilon}{}^{\alpha_n} t'_n \equiv s$$

where $\alpha_0 + 1 + \alpha_1 + 1 + \alpha_2 + 1 + \dots + \alpha_{n-1} + 1 + \alpha_n = \alpha$. Note that

$$\alpha_0 + \dots + \alpha_{n-1} < \alpha$$

and thus compressible by the induction hypothesis. So, using Lemma 47, we can find u such that $t \twoheadrightarrow u \to_{\backsim\epsilon}{}^{\leqslant\alpha} s$ as depicted in Figure 11. Then, we have $u(\epsilon) = s(\epsilon)$ and $u/i \to^{\leqslant\alpha} s/i$ for every applicable i. Similarly, we can find u'_i such that $u/i \twoheadrightarrow u'_i \to_{\backsim\epsilon}{}^{\leqslant\alpha} s/i$ for each i. Note that every $u/i \twoheadrightarrow u'_i$ is parallel to one another, and thus we can construct a convergent reduction sequence $t \twoheadrightarrow u \twoheadrightarrow \dots$ leading to s, by iterating this argument. □

Lemma 48 (Projection Lemma, or Parallel Moves Lemma) *In an orthogonal term rewriting system, if* $t \twoheadrightarrow s$ *and* $t \to t'$*, then there exists* $u \in \mathcal{T}$ *such that* $s \twoheadrightarrow u$ *and* $t' \twoheadrightarrow u$*.*

Proof: By orthogonality of the system, the redex of the reduction $t \to t'$ can be collapsed or copied, but never disturbed during the reduction

$$t \equiv t_0 \to t_1 \to \dots \to t_n \equiv s$$

Thus, we can use the rule which is applied in $t \to t'$ on all the copied redexes in t_i, the positions of which are all parallel, to obtain the diagram shown in Figure 12. We refer to [46, Chapter 4] for a more detailed explanation. □

Lemma 49 *In an orthogonal term rewriting system, if* $t \twoheadrightarrow s$ *and* $t \twoheadrightarrow s'$*, then there exists* $u \in \mathcal{T}$ *such that* $s \twoheadrightarrow u$ *and* $s' \twoheadrightarrow u$*, viz. every orthogonal system has the finitary confluence property.*

Proof: By mathematical induction on the length of the reduction $t \twoheadrightarrow s'$. Apply the above lemma along the reduction $t \twoheadrightarrow s'$ to project the reduction $t \twoheadrightarrow s$ to $s' \twoheadrightarrow u$ (see Figure 13). □

Lemma 50 *In an orthogonal infinitary term rewriting system, if* $t \twoheadrightarrow s$, $t \twoheadrightarrow s'$, *and* $s, s' \in \mathcal{N}$, *then* $s \equiv s'$, *viz. every orthogonal system has the unique infinitary normal form property.*

Proof: For a proof by contradiction we assume $t \twoheadrightarrow s$, $t \twoheadrightarrow s'$, $s, s' \in \mathcal{N}$ and $s \not\equiv s'$ in an orthogonal system. We can find $p \in \mathrm{dom}(s) \cap \mathrm{dom}(s)'$ such that $s(p) \neq s'(p)$. Since the lefthand-side of every rule is finite, we can find $s_0, s_0' \in \mathcal{T}$ such that $t \twoheadrightarrow s_0 \twoheadrightarrow s$ and $t \twoheadrightarrow s_0' \twoheadrightarrow s'$, where any reduction beginning with s_0 or s_0' is strongly p-stable, by strong convergence of each reduction, using the Compression Lemma. Thus, s_0 and s_0' can have no common reduct, which condradicts the finite confluence of the system guaranteed by Lemma 49. □

Lemma 51 *In an orthogonal infinitary term rewriting system, let* T *be a subset of* $\mathcal{T}$ *closed under subterms and reduction, i.e.*

1. *If* $t \in \mathsf{T}$, *then* $t/p \in \mathsf{T}$ *for every* $p \in \mathrm{dom}(t)$.
2. *If* $t \in \mathsf{T}$ *and* $t \twoheadrightarrow s$, *then* $s \in \mathsf{T}$.

Then, the following conditions are all equivalent:

1. *For every* $t \in \mathsf{T}$, *there exists* $s \in \mathcal{N}$ *such that* $t \twoheadrightarrow s$, *viz. the system is weakly infinitarily normalizing in* T.
2. *For any* $t \in \mathsf{T}$, *there exists no infinite root-active reduction beginning with* t, *i.e. for any infinite reduction sequence*

$$t \equiv t_0 \to_{p_0} t_1 \to_{p_1} t_2 \to_{p_2} \cdots$$

   there are only finitely many i *satisfying* $p_i = \epsilon$.

3. *For any* $t \in \mathsf{T}$, *there exists no divergent reduction sequence beginning with* t, *viz. the system is strongly infinitarily normalizing in* T.

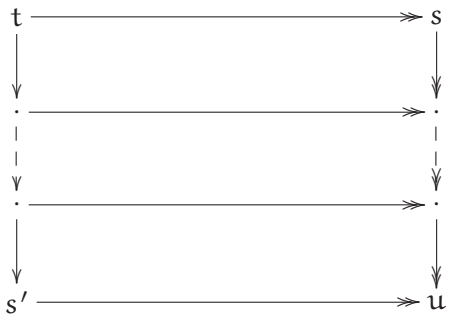


Figure 13: *Finitary confluence*

Proof: (1⇒2) For a proof by contradiction, assume that t has an infinitary normal form $t \twoheadrightarrow s \in \mathcal{N}$, and an infinite root-active reduction sequence $t \equiv t_0 \to t_1 \to \ldots$. Similarly as in the proof of Lemma 50, we can find t' such that $t \twoheadrightarrow t' \to^{\omega} s$ and that any reduction beginning with t' is strongly ϵ-stable. Now, project the reduction $t \twoheadrightarrow t'$ along the infinite root-active reduction sequence $t \equiv t_0 \to t_1 \to \ldots$ to obtain an infinite root-active reduction sequence beginning with t', which contradicts strong ϵ-stability. For further explanation we refer to [33, Section 6].

(2⇒3) To construct an infinite root-active reduction sequence from a divergent reduction sequence will suffice. Let $t_0 \to t_1 \to t_2 \to \ldots$ be a divergent reduction sequence. Then, we can find $p \in \mathbf{N}_+^*$ such that the sequence contains infinitely many redexes at the position p, and that the sequence is eventually weakly p-stable, i.e. $t_n \to t_{n+1} \to \ldots$ is weakly p-stable. Now, choose such an n that $t_n \to t_{n+1} \to \ldots$ is weakly p-stable, and let $s_i = t_{n+i}/p$. Then, for every $i \in \mathbf{N}$, we have $s_i \to_q s_{i+1}$ if $t_{n+i} \to_{p \cdot q} t_n$, or otherwise $s_i \equiv s_{i+1}$. Since the sequence $t_n \to t_{n+1} \to \ldots$ contains infinitely many steps at the redex position p, we have $s_i \to_\epsilon s_{i+1}$ for infinitely many i. Collapse the adjacent identical terms from the sequence $\langle s_i \rangle$ to obtain an infinite root-active reduction sequence.

(3⇒1) In infinitary rewriting, we can prolong the reduction sequence 'infinitarily', until we meet either an infinitary normal form or a divergent reduction sequence. □

Corollary 52 *An algorithmic term rewriting system is* WN *if and only if there exists no infinite root-active reduction sequence beginning with a proper ground term.* □

Corollary 53 *An algorithmic term rewriting system is* WN *if and only if the system is strongly infinitarily normalizing with respect to proper terms.*

Proof: The implication of the direction 'if' is trivial. Now, suppose the system is not strongly infinitarily normalizing with respect to proper terms. Then, by the above lemma, we can find $t \in \mathcal{T}^\star$ having no infinitary normal form. By non-emptiness of the algorithmic system, we can choose $v_S \in \mathcal{V}^\star$ of the sort S, for each $S \in \mathcal{S}$. Let $\sigma(x) = v_{\mathrm{out}(x)}$ for every $x \in \mathcal{X}$. Then, $t\sigma$ can have no infinitary normal form, by functionality of the algorithmic system. Hence, the system is not WN. □

Chapter 2

Technical preliminaries

In this chapter, we present examples of sorts and algorithmic systems that will be used throughout the thesis. Moreover, we introduce some notions: quasiorders, dual-compatibility, inductive height, and algebraic interpretation. Section 2.1 shows examples. Section 2.2 recalls the notion of quasiorder. Well-founded quasiorders are a quite useful to describe 'those which will terminate in finite time'. To ensure that something can happen only for finitely many times, we wish to project 'events' on to well-founded quasiorders. Thus, the notion of dual-compatibility arises, in Section 2.3.

Next, we wish to introduce the device of algebraic interpretation on proper terms. However, because of the infinite structure of proper terms, a careful treatment is needed to extend the domain of algebraic interpretation from finite terms to proper terms. To that end, in Section 2.4 we introduce the notion of inductive height, and in Section 2.5 we formalize the extended version of algebraic interpretation. Section 2.6 gives a scheme for denotational semantics of the algorithmic system as the sorted version of algebraic interpretation. There we also mention adequacy of the semantics.

2.1 Examples

We present some examples of algorithmic term rewriting systems, which will be used as running examples throughout the following chapters.

2.1.1 Sorts and constructors

First, we present sorts and constructors.

1. Boolean values, of sort BOOL. A pair of nullary constructors

$$\mathsf{T} : () \to \mathsf{BOOL}$$
$$\mathsf{F} : () \to \mathsf{BOOL}$$

have the output sort BOOL. This is an inductive sort.

$$\mathsf{BOOL} :\overset{\mathbf{I}}{=} \mathsf{T} \mid \mathsf{F}$$

2. Natural numbers, of sort NAT. The constructors

$$\begin{aligned} \mathsf{0} :&\quad () \to \mathsf{NAT} \\ \mathsf{s} :&\ \mathsf{NAT} \to \mathsf{NAT} \end{aligned}$$

have the output sort NAT. This is an inductive sort.

$$\mathsf{NAT} :\overset{\mathbf{I}}{=} \mathsf{0} \mid \mathsf{s}(\mathsf{NAT})$$

3. Lists, of sort LIST_S, representing finite sequences of objects of sort S, for any sort S. Lists of S are implemented by the empty list

$$[]^S : () \to \mathsf{LIST}_S$$

and the concatenated list of an object and a list,

$$\mathsf{cons}^S_{\mathsf{fin}} : S \times \mathsf{LIST}_S \to \mathsf{LIST}_S$$

Each LIST_S is an inductive sort.

$$\mathsf{LIST}_S :\overset{\mathbf{I}}{=} []^S \mid \mathsf{cons}^S_{\mathsf{fin}}(S, \mathsf{LIST}_S) \qquad (S \in \mathcal{S})$$

For convenience, we write $x; y$ to denote $\mathsf{cons}^S_{\mathsf{fin}}(x, y)$, for *every* S. Moreover, we write $[]$ to mean any $[]^S$ when the sort is clear from the context. For example, $\mathsf{T}; \mathsf{F}; []$ stands for

$$\mathsf{cons}^{\mathsf{BOOL}}_{\mathsf{fin}}(\mathsf{T}, \mathsf{cons}^{\mathsf{BOOL}}_{\mathsf{fin}}(\mathsf{F}, []^{\mathsf{BOOL}}))$$

and $\mathsf{0}; \mathsf{s}\,\mathsf{0}; \mathsf{s}\,\mathsf{s}\,\mathsf{0}; []$ stands for

$$\mathsf{cons}^{\mathsf{NAT}}_{\mathsf{fin}}(\mathsf{0}, \mathsf{cons}^{\mathsf{NAT}}_{\mathsf{fin}}(\mathsf{s}\,\mathsf{0}, \mathsf{cons}^{\mathsf{NAT}}_{\mathsf{fin}}(\mathsf{s}\,\mathsf{s}\,\mathsf{0}, []^{\mathsf{NAT}})))$$

4. Streams, of sort STREAM_S, representing infinite sequences of objects of sort S, for any sort S. Streams of S are implemented by the constructor

$$\mathsf{cons}^S : S \times \mathsf{STREAM}_S \to \mathsf{STREAM}_S$$

For each sort S, the sort STREAM_S is a coinductive sort.

$$\mathsf{STREAM}_S :\overset{\mathbf{C}}{=} \mathsf{cons}^S(S, \mathsf{STREAM}_S) \qquad (S \in \mathcal{S})$$

We write $x : y$ to denote $\mathsf{cons}^S(x, y)$, for every S. It should be remarked that the only constructor cons^S whose output sort is STREAM_S has an input sort STREAM_S itself at the second argument. So, if this was an inductive sort, then STREAM_S would be an empty sort, i.e. no object could have sort STREAM_S. As it is, STREAM_S is non-empty, and all the instances are inherently infinite as a term.

5. Countable ordinals, of sort ORD. The constructors

$$\begin{aligned} \mathsf{O} : &\quad () \to \mathsf{ORD} \\ \mathsf{S} : &\quad \mathsf{ORD} \to \mathsf{ORD} \\ \mathsf{L} : &\quad \mathsf{STREAM}_{\mathsf{ORD}} \to \mathsf{ORD} \end{aligned}$$

have the output sort ORD. This is an inductive sort, but the construction of objects depends on a coinductive sort $\mathsf{STREAM}_{\mathsf{ORD}}$.

$$\mathsf{ORD} :\overset{\mathbf{I}}{=} \mathsf{O} \mid \mathsf{S}(\mathsf{ORD}) \mid \mathsf{L}(\mathsf{STREAM}_{\mathsf{ORD}})$$

6. Coinductive natural numbers, of the coinductive sort CONAT, representing $\mathbf{N} \cup \{\infty\}$ with $\infty + n = n + \infty = \infty$ for all $n \in \mathbf{N} \cup \{\infty\}$. These numbers can be implemented by the constructors $\dot{0} : () \to \mathsf{CONAT}$ and $\dot{s} : \mathsf{CONAT} \to \mathsf{CONAT}$.

$$\mathsf{CONAT} :\overset{\mathbf{C}}{=} \dot{0} \mid \dot{s}(\mathsf{CONAT})$$

Though this sort does not appear in the running examples, we present this to contrast with the inductive sort NAT. The term $\mathsf{sss}\ldots$ is not proper, but $\dot{s}\dot{s}\dot{s}\ldots$ is. This $\dot{s}\dot{s}\dot{s}\ldots$ represents ∞.

2.1.2 Some algorithmic systems

The above examples introduce the 'static' part of an algorithmic system, i.e. the objects only; we now give some examples including moreover the 'dynamic' part, i.e. the rewrite rules.

Hamming numbers

Following most of tutorials on functional programming, we first implement the enumeration of the Hamming numbers as $\mathsf{STREAM}_{\mathsf{NAT}}$.

A Hamming number is a natural number whose prime factors consists of 2, 3, and 5 only. We intend to enumerate these numbers in ascending order.

$$\begin{aligned} \mathsf{Ham} &\to \mathsf{s}\,0 : \mathsf{mrg}(\mathsf{mrg}(\mathsf{Ham2}, \mathsf{Ham3}), \mathsf{Ham5}) \\ \mathsf{Ham2} &\to \mathsf{sca}(\mathsf{Ham}, \mathsf{s}\,\mathsf{s}\,0) \\ \mathsf{Ham3} &\to \mathsf{sca}(\mathsf{Ham}, \mathsf{s}\,\mathsf{s}\,\mathsf{s}\,0) \\ \mathsf{Ham5} &\to \mathsf{sca}(\mathsf{Ham}, \mathsf{s}\,\mathsf{s}\,\mathsf{s}\,\mathsf{s}\,\mathsf{s}\,0) \end{aligned}$$

will provide the enumeration of the Hamming numbers with Ham, where the function mrg enumerates the union of a pair of enumerated streams of natural numbers; the function sca computes the scalar multiple, or pointwise multiplication, of a stream by a natural number. Thus, we have to

implement mrg and sca. The function mrg will mutually depend on aux, an auxiliary function to implement mrg, and the comparison function cmp. The function sca will depend on multiplication on natural numbers; multiplication depends on addition.

The whole well-known system of HAM is as in Table 14.

$$
\begin{aligned}
n + 0 &\to n \\
n + s\,m &\to s(n + m) \\
n \times 0 &\to 0 \\
n \times s\,m &\to (n \times m) + n \\
\mathsf{sca}(n : x, m) &\to (n \times m) : \mathsf{sca}(x, m) \\
\mathsf{cmp}(0, 0) &\to \mathsf{eq} \\
\mathsf{cmp}(s\,n, 0) &\to \mathsf{gt} \\
\mathsf{cmp}(0, s\,m) &\to \mathsf{lt} \\
\mathsf{cmp}(s\,n, s\,m) &\to \mathsf{cmp}(n, m) \\
\mathsf{mrg}(n : x, m : y) &\to \mathsf{aux}(\mathsf{cmp}(n, m), n : x, m : y) \\
\mathsf{aux}(\mathsf{eq}, n : x, m : y) &\to n : \mathsf{mrg}(x, y) \\
\mathsf{aux}(\mathsf{gt}, x, m : y) &\to m : \mathsf{mrg}(x, y) \\
\mathsf{aux}(\mathsf{lt}, n : x, y) &\to n : \mathsf{mrg}(x, y) \\
\mathsf{Ham} &\to s\,0 : \mathsf{mrg}(\mathsf{mrg}(\mathsf{Ham2}, \mathsf{Ham3}), \mathsf{Ham5}) \\
\mathsf{Ham2} &\to \mathsf{sca}(\mathsf{Ham}, s\,s\,0) \\
\mathsf{Ham3} &\to \mathsf{sca}(\mathsf{Ham}, s\,s\,s\,0) \\
\mathsf{Ham5} &\to \mathsf{sca}(\mathsf{Ham}, s\,s\,s\,s\,s\,0)
\end{aligned}
$$

Table 14: *The system* HAM

Tree ordinals

Next, we implement the representation of countable ordinals in the framework of tree ordinals, as in Subsection 0.4.2. As seen in Subsection 2.1.1, we have the constructor O to represent 0; the constructor S to represent successor ordinals, and the constructor L for suprema, mainly used to represent limit ordinals. Thus, given a representation of α, we can represent $\alpha + \omega$ as $\mathsf{L}(\mathsf{nats}(\ulcorner \alpha \urcorner))$ with help of the auxiliary function $\mathsf{nats} : \mathsf{ORD} \to \mathsf{STREAM}_{\mathsf{ORD}}$ given as

$$\mathsf{nats}(n) \to n : \mathsf{nats}(\mathsf{S}\,n)$$

Here we implement basic arithmetic operations on tree ordinals: addition, multiplication, and exponentiation. We stipulate $0^0 = 0$ for conve-

$$
\begin{aligned}
n + \mathsf{O} &\to n \\
n + \mathsf{S}\, m &\to \mathsf{S}(n + m) \\
n + \mathsf{L}\, x &\to \mathsf{L}(\mathsf{add}_{\mathsf{L}}(n, x)) \\
\mathsf{add}_{\mathsf{L}}(n, m : x) &\to (n + m) : \mathsf{add}_{\mathsf{L}}(n, x) \\
n \cdot \mathsf{O} &\to \mathsf{O} \\
n \cdot \mathsf{S}\, m &\to (n \cdot m) + n \\
n \cdot \mathsf{L}\, x &\to \mathsf{L}(\mathsf{mul}_{\mathsf{L}}(n, x)) \\
\mathsf{mul}_{\mathsf{L}}(n, m : x) &\to (n \cdot m) : \mathsf{mul}_{\mathsf{L}}(n, x) \\
\mathsf{exp}(\mathsf{O}, \mathsf{O}) &\to \mathsf{O} \\
\mathsf{exp}(\mathsf{S}\, n, \mathsf{O}) &\to \mathsf{S}\, \mathsf{O} \\
\mathsf{exp}(\mathsf{L}\, x, \mathsf{O}) &\to \mathsf{L}(\mathsf{pow0}_{\mathsf{L}}(x)) \\
\mathsf{exp}(n, \mathsf{S}\, m) &\to \mathsf{exp}(n, m) \cdot n \\
\mathsf{exp}(n, \mathsf{L}\, x) &\to \mathsf{L}(\mathsf{exp}_{\mathsf{L}}(n, x)) \\
\mathsf{pow0}_{\mathsf{L}}(n : x) &\to \mathsf{exp}(n, \mathsf{O}) : \mathsf{pow0}_{\mathsf{L}}(x) \\
\mathsf{exp}_{\mathsf{L}}(n, m : x) &\to \mathsf{exp}(n, m) : \mathsf{exp}_{\mathsf{L}}(n, x) \\
\mathsf{omega} &\to \mathsf{L}(\mathsf{nats}(\mathsf{O})) \\
\mathsf{nats}(n) &\to n : \mathsf{nats}(\mathsf{S}\, n)
\end{aligned}
$$

Table 15: *The rewrite rules of the system* ORD-AME

nience, so that exponentiation can be regarded as a total binary function, just as addition and multiplication. Each of these operations is given to be continuous in its second argument, i.e.

$$
\begin{aligned}
a + \sup B &= \sup_{b \in B}(a + b) \\
a \cdot \sup B &= \sup_{b \in B}(a \cdot b) \\
a^{\sup B} &= \sup_{b \in B}(a^b)
\end{aligned}
$$

Thus, the rewriting rules for multiplication, for example, are as follows.

$$
\begin{aligned}
n \cdot \mathsf{O} &\to \mathsf{O} \\
n \cdot \mathsf{S}\, m &\to (n \cdot m) + n \\
n \cdot \mathsf{L}\, x &\to \mathsf{L}(\mathsf{mul}_{\mathsf{L}}(n, x)) \\
\mathsf{mul}_{\mathsf{L}}(n, m : x) &\to (n \cdot m) : \mathsf{mul}_{\mathsf{L}}(n, x)
\end{aligned}
$$

So, the rewrite rules of the algorithmic system of ORD-AME are as in Table 15. The acronym 'AME' is for addition, multiplication, and

exponentiation. Note that the case distinction for exponentiation is a bit awkward because of 0^0; one might think of letting $0^0 = 1$ for a simpler implementation, however, that system would not be able to compute $0^\omega = 0$ correctly.

For example, $2 \cdot \omega$ is computed as follows.

$$\begin{aligned}
\ulcorner 2 \urcorner \cdot \ulcorner \omega \urcorner &\equiv \ulcorner 2 \urcorner \cdot \mathsf{omega} \\
&\twoheadrightarrow \ulcorner 2 \urcorner \cdot \mathsf{L}(\ulcorner 0 \urcorner : \ulcorner 1 \urcorner : \ulcorner 2 \urcorner : \ulcorner 3 \urcorner : \dots) \\
&\to \mathsf{L}(\mathsf{mul}_{\mathsf{L}}(\ulcorner 2 \urcorner, \ulcorner 0 \urcorner : \ulcorner 1 \urcorner : \ulcorner 2 \urcorner : \ulcorner 3 \urcorner : \dots)) \\
&\twoheadrightarrow \mathsf{L}((\ulcorner 2 \urcorner \cdot \ulcorner 0 \urcorner) : (\ulcorner 2 \urcorner \cdot \ulcorner 1 \urcorner) : (\ulcorner 2 \urcorner \cdot \ulcorner 2 \urcorner) : (\ulcorner 2 \urcorner \cdot \ulcorner 3 \urcorner) : \dots) \\
&\twoheadrightarrow \mathsf{L}(\ulcorner 0 \urcorner : \ulcorner 2 \urcorner : \ulcorner 4 \urcorner : \ulcorner 6 \urcorner : \dots)
\end{aligned}$$

Since $\sup\{0, 2, 4, 6, \dots\} = \omega$, the result is another representation of ω. Hence, $2 \cdot \omega = \omega$.

On the other hand, $\omega \cdot 2$ goes as follows:

$$\begin{aligned}
\ulcorner \omega \urcorner \cdot \ulcorner 2 \urcorner &\equiv \mathsf{omega} \cdot \mathsf{S}\,\mathsf{S}\,\mathsf{O} \\
&\to (\mathsf{omega} \cdot \mathsf{S}\,\mathsf{O}) + \mathsf{omega} \\
&\to ((\mathsf{omega} \cdot \mathsf{O}) + \mathsf{omega}) + \mathsf{omega} \\
&\to (\mathsf{O} + \mathsf{omega}) + \mathsf{omega} \\
&\twoheadrightarrow (\ulcorner \omega \urcorner) + \mathsf{omega} \\
&\twoheadrightarrow \mathsf{L}(\ulcorner \omega \urcorner : \ulcorner \omega + 1 \urcorner : \ulcorner \omega + 2 \urcorner : \ulcorner \omega + 3 \urcorner : \dots)
\end{aligned}$$

2.2 Quasiorders

The WN and DN properties of a system and properness of a term can be phrased as '*something* cannot eventually always happen'. Thus, the notion of a well-founded quasiordering of terms will play an important role in dealing with those properties.

Definition 54 (quasiorders) Let A be a set. A binary relation $R \subseteq A \times A$ is a *quasiorder* if it is reflexive and transitive, i.e. $a\ R\ a$ for every $a \in A$ and $a\ R\ b,\ b\ R\ c \Rightarrow a\ R\ c$ for every $a, b, c \in A$.

We will use $\preccurlyeq$ to denote a quasiorder, occasionally with a subscript. Given a quasiorder $\preccurlyeq$, we use $\prec$, $\succcurlyeq$, and $\succ$ defined as

$$\begin{aligned}
a \prec b &\Leftrightarrow a \preccurlyeq b \text{ and } b \not\preccurlyeq a \\
a \succcurlyeq b &\Leftrightarrow b \preccurlyeq a \\
a \succ b &\Leftrightarrow b \prec a.
\end{aligned}$$

A quasiorder $\preccurlyeq$ is *well-founded* if there exists no infinite descending chain $a_0 \succ a_1 \succ a_2 \succ \dots$. ◻

Lemma 55 (induction principle) *On a set A with a well-founded quasiorder $\preccurlyeq$, we can perform generalized mathematical induction. That is, if a subset $B \subseteq A$ satisfies the induction clause*

$$\forall a \in A.\ (\forall b \in A.\ b \prec a \Rightarrow b \in B) \Rightarrow a \in B$$

then $B = A$.

Proof: For a proof by contradiction, we assume that B satisfies the above induction clause and $B \neq A$. We will construct an infinite descending chain $a_0 \succ a_1 \succ a_2 \succ \ldots$ such that $a_i \in A \setminus C$.

Let $C = A \setminus B$. Since $B \subseteq A$ and $B \neq A$, we have $C \neq \varnothing$ and thus we can find $a \in C$. Let $a_0 = a$.

Suppose $a_n \in C$ is already defined. Let $C_n = \{c \in C \mid c \prec a_n\}$. If $C_n = \varnothing$, then, letting $a = a_n$ in the induction hypothesis, we have $a_n \in B$, conflicting with the assumption $a_n \in C$. Thus, we can choose $a_{n+1} \in C_n$. From the definition of C_n, we have $a_{n+1} \in C$ and $a_n \succ a_{n+1}$.

Thus, we can construct an infinite descending chain $a_0 \succ a_1 \succ a_2 \succ \ldots$ in A, contradicting the well-foundedness of $\preccurlyeq$. □

Lemma 56 *Given a set A and a well-founded quasiorder $\preccurlyeq$ on A, for each $\alpha \in \mathbf{On}$, let A_α be a subset of A defined by transfinite induction on α as follows:*

$$A_\alpha = \{a \in A \mid \forall b \in A.\ b \prec a \Rightarrow b \in \bigcup_{\iota < \alpha} A_\iota\}$$

Then, we have $A = \bigcup_{\alpha \in \mathbf{On}} A_\alpha$.

Proof: By induction on $\preccurlyeq$. Let $A_{\mathbf{On}} = \bigcup_{\alpha \in \mathbf{On}} A_\alpha$. Note that $\alpha \leqslant \beta$ implies $A_\alpha \subseteq A_\beta$.

Let a be an arbitrary element in S and suppose that, for every $b \in A$, $b \prec a$ implies $b \in A_{\mathbf{On}}$. To show $a \in A_{\mathbf{On}}$ will suffice (by Lemma 55). Let $B_a = \{b \in A \mid b \prec a\}$. By assumption, we have $B_a \subseteq A_{\mathbf{On}}$. Thus, for each $b \in B_a$, we can find $\alpha_b \in \mathbf{On}$ such that $b \in A_{\alpha_b}$. Let $\alpha = \sup_{b \in B_a}\{\alpha_b\}$. Since we have $\alpha_b \in A_{\alpha_b} \subseteq A_\alpha$ for every B_a, from the definition of $A_{(-)}$ we have $a \in A_\alpha$. Hence, $a \in A_{\mathbf{On}}$. □

Definition 57 Given a set A and a well-founded quasiorder $\preccurlyeq$ on A, we define $\mathrm{rank}_\preccurlyeq : A \to \mathbf{On}$ given by

$$\mathrm{rank}_\preccurlyeq(a) = \min\{\alpha \in \mathbf{On} \mid a \in A_\alpha\}$$

where A_α is as given in the above lemma. ◪

Definition 58

1. Let $A_1, \ldots, A_n, B$ be quasiordered sets. Then, a function

$$f : A_1 \times \cdots \times A_n \to B$$

is *(weakly) monotonic* if $a_i \preccurlyeq a'_i$ for $1 \leqslant i \leqslant n$ implies

$$f(a_1, \ldots, a_n) \preccurlyeq f(a'_1, \ldots, a'_n)$$

2. An n-ary function f on a quasiordered set A is *strongly increasing* if $f(a_1, \ldots, a_n) > a_i$ for every $1 \leqslant i \leqslant n$. ◻

2.3 Compatibility

Finitary termination can be ensured by finding a well-founded quasiorder $\preccurlyeq$ on terms such that $t \succ s$ whenever $t \to s$. Temporarily we call such quasiorders 'strongly $\to$-compatible'. Conversely, every finitarily terminating system admits a strongly $\to$-compatible well-founded quasiorder. Finitary termination of the system thus can be characterized by a strongly $\to$-compatible well-founded quasiorder. Most of the analysis of finitary termination is directly or indirectly based on this characterization.

In the analysis of infinitary normalization, we will see in the next chapter that an algorithmic term rewriting system is WN if and only if there exists no root-active infinite reduction sequence

$$t_0 \to t_1 \to t_2 \to_\epsilon t_3 \to t_4 \to_\epsilon \ldots$$

beginning with a proper ground term, containing infinitely many root reductions $\to_\epsilon$. In order to characterize the nonexistence of such a sequence, we wish to find a well-founded quasiorder and translate the above sequence to an infinite descending chain. In the above example on finitary termination, a non-terminating sequence $t_0 \to t_1 \to t_2 \to \ldots$ is translated to an infinite descending chain $t_0 \succ t_1 \succ t_2 \succ \ldots$ by a strongly $\to$-compatible quasiorder. In the present case, since we allow infinitely long reductions, strong $\to$-compatibility is too strong to characterize the WN property. So, we think of replacing root reductions $\to_\epsilon$ by $\succ$, and the other lower-level reductions by $\succcurlyeq$. Thus, an appropriate quasiorder should satisfy the following conditions:

1. If $t \to_\epsilon s$, then $t \succ s$. This property is temporarily called 'strongly $\to_\epsilon$-compatible'.

2. If $t \to s$, then $t \succcurlyeq s$. This property is temporarily called 'weakly $\to$-compatible'.

3. The quasiorder is well-founded.

In fact, we will see that an algorithmic term rewriting system is WN if and only if there exists a strongly $\to_\epsilon$-compatible and weakly $\to$-compatible well-founded quasiorder on initially proper ground terms (Theorems 96 and 97).

For convenience, we introduce a 'dual-compatibility notation' as follows.

Definition 59 (dual-compatibility) Let R and R' be binary relations on a set A. Then, a quasiorder $\preccurlyeq$ on A is $R \,|\, R'$-*compatible* if for every $a, b \in A$, $a \mathrel{R} b$ implies $a \succ b$ and $a \mathrel{R'} b$ implies $a \succcurlyeq b$. In this notation, we will omit the symbol $\cup$; for example, we write "$R_1 R_2 R_3 \,|\, R'_1 R'_2$-compatible" to denote $(R_1 \cup R_2 \cup R_3) \,|\, (R'_1 \cup R'_2)$-compatibility. ◿

The following proposition is immediate.

Proposition 60 *A quasiorder is* $R_1 \ldots R_n \,|\, R'_1 \ldots R'_m$*-compatible if and only if the quasiorder is* $R_i \,|\, \varnothing$*-compatible for every* $i = 1, \ldots, n$ *and* $\varnothing \,|\, R'_j$*-compatible for every* $j = 1, \ldots, m$. □

2.4 Inductive height

In this section we define the notion of inductive height. This notion arises when we characterise properness of a term t by a well-founded quasiorder on $\mathrm{dom}(t)$. First, we define the *immediate subterm relation* as follows.

Definition 61 Let t be a term. Then, for every $i = 1, \ldots, \mathrm{ar}(t(\epsilon))$, we write $t \searrow t/i$. Moreover, we divide the relation $\searrow$ into three kinds:

1. Defined function application. We write $t \searrow^{\mathcal{D}} s$ if $t(\epsilon) \in \mathcal{D}$.
2. Inductive construction. We write $t \searrow^{\mathbf{I}} s$ if $t(\epsilon) \in \mathcal{C}$ and $\mathrm{srt}(s) \in \mathcal{S}^{\mathbf{I}}$.
3. Coinductive construction. We write $t \searrow^{\mathbf{C}} s$ if $t(\epsilon) \in \mathcal{C}$ and $\mathrm{srt}(s) \in \mathcal{S}^{\mathbf{C}}$.

Note that whenever $t \searrow s$, one of $\searrow^{\mathcal{D}}$, $\searrow^{\mathbf{I}}$, or $\searrow^{\mathbf{C}}$ holds between t and s. ◿

Then, a path in a term t can be specified as an infinite sequence of immediate subterming:

$$t \equiv t_0 \searrow t_1 \searrow t_2 \searrow \ldots$$

If there occur infinitely many $\searrow^{\mathcal{D}}$- or $\searrow^{\mathbf{I}}$-steps in the above sequence, the path is vicious. Hence, for proper terms, there exists some ordinal assignment $\theta : \mathcal{T}^\star \to \mathbf{On}$ such that $t \searrow s$ implies $\theta(t) \geqslant \theta(s)$, and that $t \searrow^{\mathcal{D}} s$ or $t \searrow^{\mathbf{I}} s$ implies $\theta(t) > \theta(s)$. We take the minimal θ and refer to $\theta(t)$ as the *inductive height of* t, written $\lceil t \rceil$. Formally:

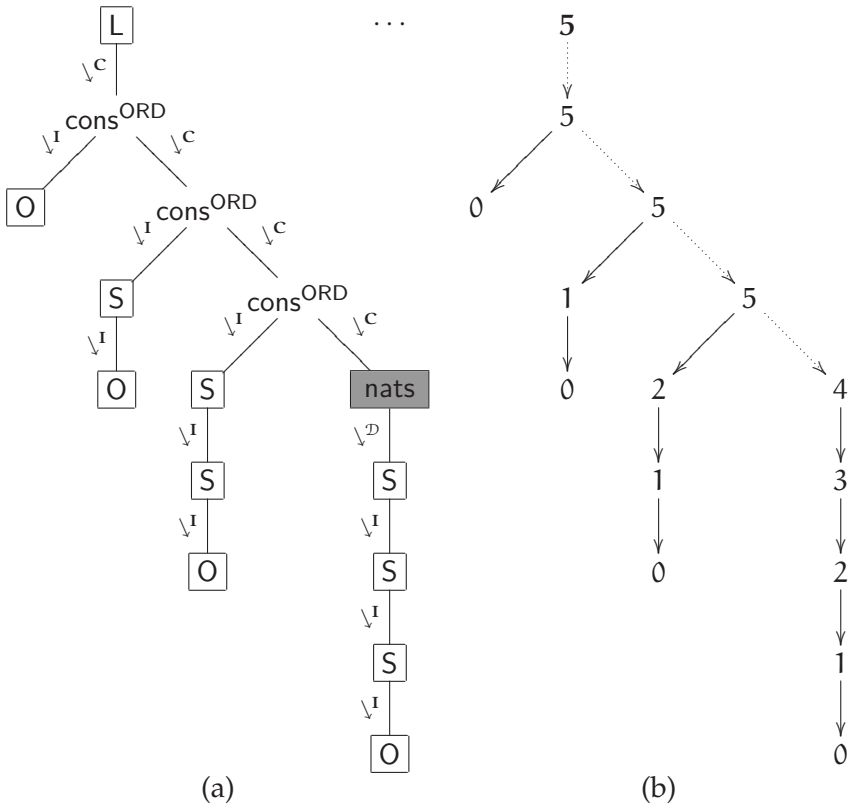


Figure 16: *Inductive height of* $\mathsf{L}(\mathsf{O} : \mathsf{S}\,\mathsf{O} : \mathsf{S}\,\mathsf{S}\,\mathsf{O} : \mathsf{nats}(\mathsf{S}\,\mathsf{S}\,\mathsf{S}\,\mathsf{O}))$

Definition 62 For each $\alpha \in \mathbf{\Omega}$, let $\mathcal{T}_\alpha \subseteq \mathcal{T}$ be defined by transfinite induction on α as

$$\mathcal{T}_\alpha = \left\{ t \in \mathcal{T} \;\middle|\; \begin{array}{l} \text{if } t \setminus \ldots \setminus s \text{ with at least one } \setminus^{\mathcal{D}} \text{ or } \setminus^{\mathbf{I}} \text{ in the sequence,} \\ \text{there exists } \beta < \alpha \text{ such that } s \in \mathcal{T}_\beta \end{array} \right\}$$

and let $\lceil t \rceil = \min\{\alpha \in \mathbf{\Omega} \mid t \in \mathcal{T}_\alpha\}$. Note that $\bigcup_{\alpha \in \mathbf{\Omega}} \mathcal{T}_\alpha = \mathcal{T}^\star$. ◿

The following proposition is immediate.

Proposition 63

1. *If* $t \setminus^{\mathbf{I}} s$ *or* $t \setminus^{\mathcal{D}} s$, *then* $\lceil t \rceil > \lceil s \rceil$.
2. *If* $t \setminus^{\mathbf{C}} s$, *then* $\lceil t \rceil \geqslant \lceil s \rceil$. □

Example 64 Consider the term $\mathsf{L}(\mathsf{O} : \mathsf{S}\,\mathsf{O} : \mathsf{S}\,\mathsf{S}\,\mathsf{O} : \mathsf{nats}(\mathsf{S}\,\mathsf{S}\,\mathsf{S}\,\mathsf{O}))$. This term is a reduct of the term omega after the reduction of three steps. Figure 16(a) depicts the tree representation of the term. Constructors of an inductive

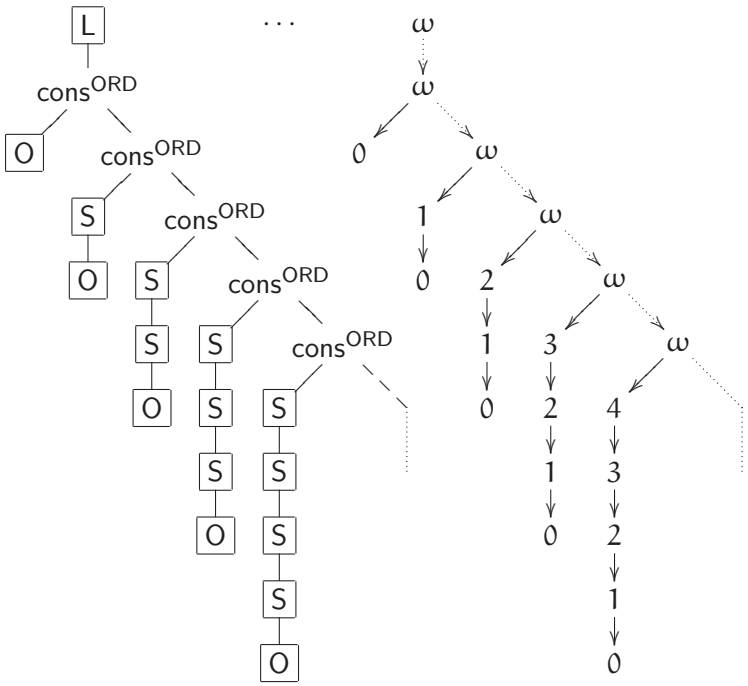


Figure 17: *Inductive height of* $\ulcorner\omega\urcorner$

sort are framed and the box of a defined symbol is drawn gray. Beside the edges, the corresponding immediate subterm relation is indicated.

As an aid to visualize the inductive height of the term, the required order between adjacent positions is as indicated in the figure (b), where solid and dotted arrows respectively denote strong and weak decrease. The labeled numbers are the minimal ordinals satisfying this order. Hence, we have $\lceil \mathsf{L}(\mathsf{O} : \mathsf{S}\,\mathsf{O} : \mathsf{S}\,\mathsf{S}\,\mathsf{O} : \mathsf{nats}(\mathsf{S}\,\mathsf{S}\,\mathsf{S}\,\mathsf{O}))\rceil = 5$. ⊡

Example 65 The inductive height of a term can be generally any countable ordinal. As an example of a term with a height greater than ω, we present the inductive height of $\ulcorner\omega \cdot 2\urcorner$. Since

$$\ulcorner\omega \cdot 2\urcorner \equiv \mathsf{L}(\ulcorner\omega\urcorner : \mathsf{S}\ulcorner\omega\urcorner : \mathsf{S}\,\mathsf{S}\ulcorner\omega\urcorner : \dots)$$

we first compute $\lceil\ulcorner\omega\urcorner\rceil$. Figure 17 shows how $\lceil\ulcorner\omega\urcorner\rceil = \omega$ is obtained. And, based on this, Figure 18 shows that $\lceil\ulcorner\omega \cdot 2\urcorner\rceil = \omega \cdot 2$. ⊡

2.5 Algebraic interpretation

The next two sections introduce first an algebraic interpretation of infinite terms, which will enable us to develop a method to prove the property DN,

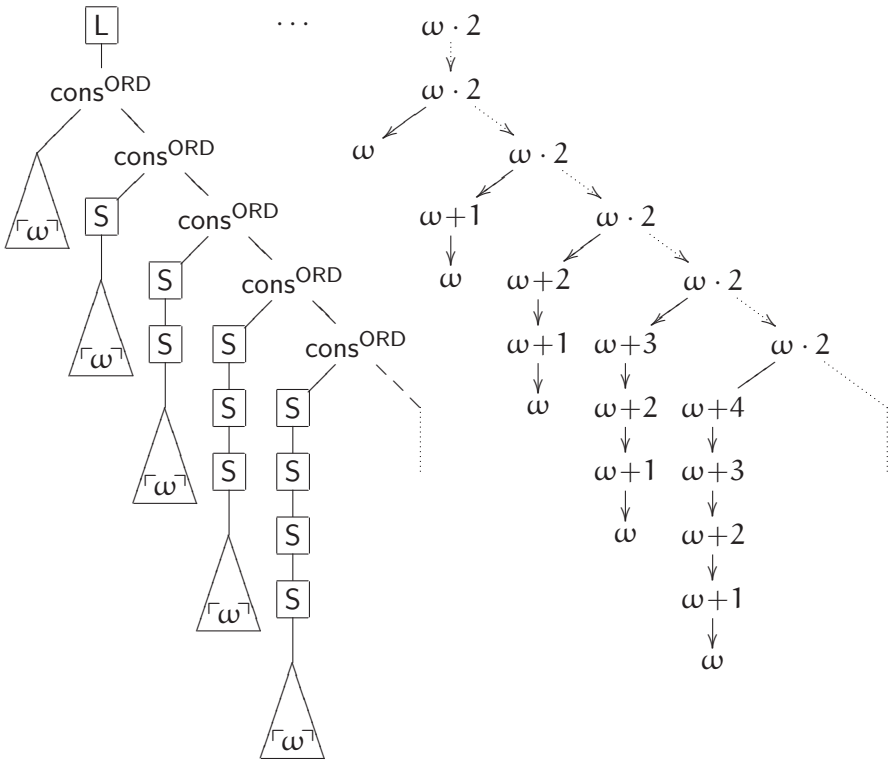


Figure 18: *Inductive height of* $\ulcorner \omega \cdot 2 \urcorner$

domain normalization, and second a 'semantical' interpretation of infinite terms, which will be employed to devise a method for proving 'adequacy' of a semantics, i.e. a semantical interpretation is preserved under transfinite reduction. These two notions, algebraic and semantical interpretation should not be confused. We briefly indicate how they are distinguished.

Conceptually, algebraic interpretation employs some algebras solely adopted and designed for a technical purpose, where the interpretation typically *decreases* in a reduction step; in such an algebra one generally does not 'recognize' the infinite terms that are interpreted. A semantical (or denotational) interpretation on the other hand, concerns often a 'natural' or 'canonical' semantics that is close to the 'actual meaning' of the terms, e.g. natural numbers, or streams, or other familiar (co-)data types. Typically, here the interpretation is *invariant* in a reduction step, or in a (possibly transfinite) reduction sequence.

A consequence of the above is that the algebraic interpretations of this section assume some order on the domain. More specifically, we will use so-called *coinductive domains* (see Definition 67), of which the prime example in our context is the partial order of countable ordinals. In contrast,

semantical interpretation employs complete metric spaces.

The method of algebraic interpretation plays an important role to ensure termination of a finitary rewriting system [46, Chapter 7]. Recently, it has also been generalized to ensure infinitary normalization in [19, 54].

We give an example where algebraic interpretation of finite terms can be used for establishing termination. Consider the system of addition and multiplication on natural numbers

$$\begin{aligned} n + 0 &\to n & n \times 0 &\to 0 \\ n + s\,m &\to s(n + m) & n \times s\,m &\to (n \times m) + n \end{aligned}$$

is shown to be terminating by the algebraic interpretation given by

$$\begin{aligned} [0] &= 0 \\ [s](a) &= a + 1 \\ [+](a, b) &= a + 2b + 1 \\ [\times](a, b) &= (a + 1)(2b + 1) \end{aligned}$$

With the above interpretation, we have

$$\begin{aligned} [n + 0] &= [n] + 1 > [n] \\ [n + s\,m] &= [n] + 2[m] + 3 > [n] + 2[m] + 2 = [s(n + m)] \\ [n \times 0] &= [n] + 1 > 0 = [0] \\ [n \times s\,m] &= ([n] + 1)(2[m] + 3) > ([n] + 1)(2[m] + 1) + 2[n] + 1 = [(n \times m) + n] \end{aligned}$$

So, whenever $t \to s$, we have $[t] > [s]$ and therefore the system is terminating.

Though not exactly in the same way, algebraic interpretation is an effective device also in analyzing productivity of algorithmic term rewriting systems. Thus, we need to extend algebraic interpretation from finite terms to possibly infinite proper terms. This will lead to the notion of a *continuous* Σ*-algebra*. We start by considering the finite case.

Definition 66 A Σ*-algebra* is a tuple $\langle A, [-]\rangle$ where A is a set and $[f]$ is an n-ary function on A for each n-ary function symbol $f \in \Sigma$. ◿

Given a Σ-algebra $\langle A, [-]\rangle$, we wish to construct a homomorphic interpretation on $\mathcal{G}^\star$, that is, a function $[-] : \mathcal{G}^\star \to A$ satisfying

$$[f(t_1, \ldots, t_n)] = [f]([t_1], \ldots, [t_n])$$

For finite ground terms, the above requirement is enough to determine the interpretation, because the above equation directly forms an inductive definition of $[-] : \mathcal{G}^\star \to A$. Since every infinite term can be regarded as the

limit of an infinite sequence of unsorted finite terms, as a first attempt to construct $[-] : \mathcal{G}^\star \to \mathcal{A}$, we think of giving it as the continuous completion:

$$[\lim_{n\to\infty} t_n] = \lim_{n\to\infty} [t_n]$$

That is possible, however, it requires $[f]$ to be continuous for *every* function symbol f. This restriction is rather strong; we wish to think of inductive height, for example, as example of an algebraic interpretation. Consider giving the interpretation functions

$$\begin{aligned} [\mathsf{O}] : \quad & \{\varnothing\} \to \boldsymbol{\Omega} \\ [\mathsf{S}] : \quad & \boldsymbol{\Omega} \to \boldsymbol{\Omega} \\ [\mathsf{L}] : \quad & \boldsymbol{\Omega} \to \boldsymbol{\Omega} \\ [\mathsf{cons}^{\mathsf{ORD}}] : \quad & \boldsymbol{\Omega} \times \boldsymbol{\Omega} \to \boldsymbol{\Omega} \end{aligned}$$

to derive the inductive height function as the algebraic interpretation. From the definition of inductive height, we have

$$\begin{aligned} [\mathsf{O}] &= 0 \\ [\mathsf{S}](\alpha) &= \alpha + 1 \\ [\mathsf{L}](\alpha) &= \alpha \\ [\mathsf{cons}^{\mathsf{ORD}}](\alpha, \beta) &= \max\{\alpha + 1, \beta\} \end{aligned}$$

First, we consider computation of $[\ulcorner \omega \urcorner]$ with the above interpretation functions. In order to get a sequence of finite terms leading to $\ulcorner \omega \urcorner$, we introduce a fresh nullary constructor symbol $\bot$ and let $[\bot]$ be 0, the least element of $\boldsymbol{\Omega}$. Now, we truncate $\ulcorner \omega \urcorner$ at depth n for each $n \in \mathbf{N}$, to get the sequence

$$\bot,\ \mathsf{L}\,\bot,\ \mathsf{L}(\bot : \bot),\ \mathsf{L}(\mathsf{O} : \bot : \bot),\ \mathsf{L}(\mathsf{O} : \mathsf{S}\,\bot : \bot : \bot),\ \ldots$$

and define

$$\begin{aligned} [\ulcorner \omega \urcorner] &= \sup\{[\bot],\ [\mathsf{L}(\bot)],\ [\mathsf{L}(\bot : \bot)],\ [\mathsf{L}(\mathsf{O} : \bot : \bot)],\ [\mathsf{L}(\mathsf{O} : \mathsf{S}\,\bot : \bot : \bot)],\ \ldots\} \\ &= \sup\{0,\ 0,\ 1,\ 1,\ 2,\ \ldots\} \end{aligned}$$

Observe that the result is certainly ω (see also Figure 19). Things seem going well with $[\ulcorner \omega \urcorner]$. Next, we try $[\ulcorner \omega + 1 \urcorner]$. Similarly, we compute the truncated terms

$$\bot,\ \mathsf{S}\,\bot,\ \mathsf{S}\,\mathsf{L}\,\bot,\ \mathsf{S}\,\mathsf{L}(\bot : \bot),\ \mathsf{S}\,\mathsf{L}(\mathsf{O} : \bot : \bot),\ \ldots$$

and define

$$\begin{aligned} [\ulcorner \omega \urcorner] &= \sup\{[\bot],\ [\mathsf{S}\,\bot],\ [\mathsf{S}\,\mathsf{L}\,\bot],\ [\mathsf{S}\,\mathsf{L}(\bot : \bot)],\ [\mathsf{S}\,\mathsf{L}(\mathsf{O} : \bot : \bot)],\ \ldots\} \\ &= \sup\{0,\ 1,\ 1,\ 2,\ 2,\ \ldots\} \end{aligned}$$

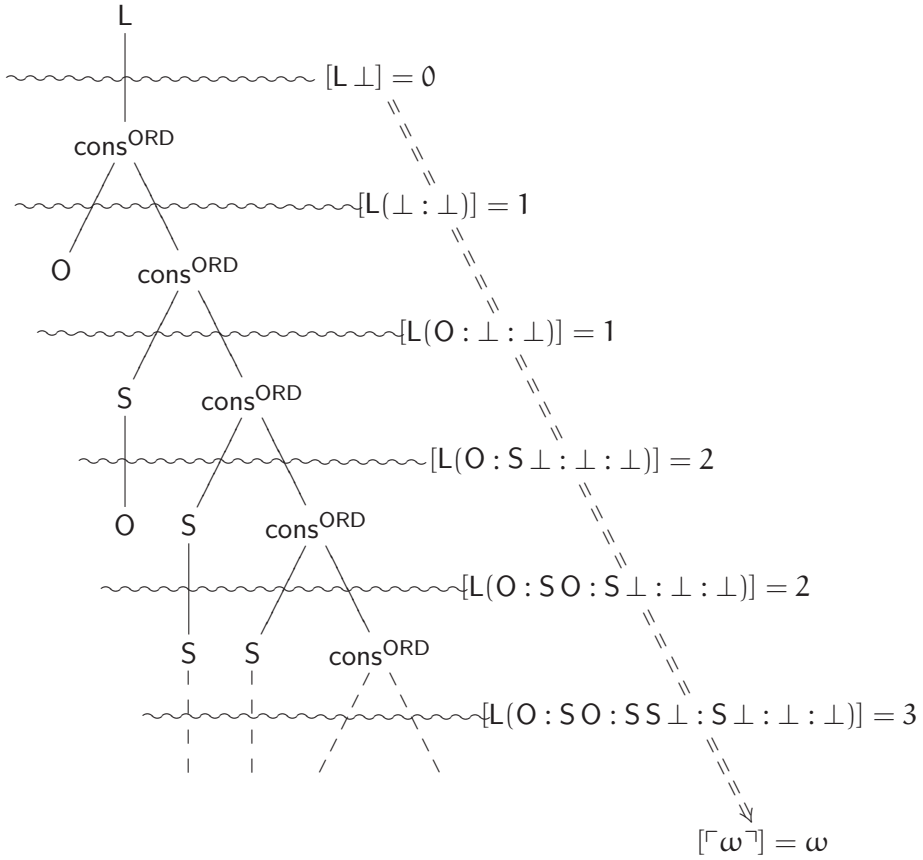


Figure 19: *Sketch of computation of* $[\ulcorner\omega\urcorner]$

The result is again ω, while the fact that $[-]$ should be homomorphic yields

$$[\ulcorner\omega + 1\urcorner] = [\mathsf{S}\ulcorner\omega\urcorner] = [\mathsf{S}]([\ulcorner\omega\urcorner]) = \omega + 1 \neq \omega$$

This trouble comes from the fact that [S] is not continuous. Of course, we wish to give priority to the homomorphism property; $[\ulcorner\omega + 1\urcorner] = \omega + 1$. The reason why we had to take the limit (or the supremum) to compute an algebraic interpretation was, because the inductive definition derived from the homomorphism property does not terminate for infinite terms. Since S cannot be infinitely nested in any proper term, the function [S] does not have to be continuous. In this example, only $\mathsf{cons}^{\mathsf{ORD}}$ can be infinitely nested at the second argument, and thus has to be continuous on the second argument.

On the basis of the above attempt, we formalize continuous Σ-algebras and derived interpretations in semantically sorted systems.

Definition 67 A *coinductive domain* is a set A where every countable subset

$B \subseteq A$ has the least upper bound $\operatorname{lub} B \in A$. Note that this condition implies the existence of the least element as $\min A = \operatorname{lub} \varnothing$. ◿

Definition 68 Let A be a coinductive domain. Then, a *continuous* Σ*-algebra* $\langle A, [-] \rangle$ consists of a function $[f] : A^n \to A$ for each n-ary function symbol f, satisfying the following condition:

> For every n-tuple of sets $B_1, \ldots, B_n \subseteq A$, if $f \in \mathcal{C}$ and B_i is singleton for every i such that $\mathrm{in}(f, i) \in \mathcal{S}^{\mathbf{I}}$, then
>
> $$\operatorname{lub}\{[f](b_1, \ldots, b_n) \mid b_i \in B_i\} = [f](\operatorname{lub} B_1, \ldots, \operatorname{lub} B_n)$$

Given a continuous Σ-algebra $\langle A, [-] \rangle$, for each ordinal $\alpha \leqslant \boldsymbol{\Omega}$, define the interpretation $[-]^\alpha : \{t \in \mathcal{G}^\star \mid \lceil t \rceil < \alpha\} \to A$ by transfinite induction on α as follows:

1. For $\alpha = 0$, the domain $\{t \in \mathcal{G}^\star \mid \lceil t \rceil < 0\}$ is empty.
2. For a successor ordinal $\alpha = \alpha' + 1$, let

$$[t]^\alpha_0 = \min A$$
$$[f(t_1, \ldots, t_n)]^\alpha_{k+1} = [f](a_1, \ldots, a_n)$$

where

$$a_i = \begin{cases} [t_i]^{\alpha'} & (\text{if } \mathrm{in}(f, i) \in \mathcal{S}^{\mathbf{I}}) \\ [t_i]^\alpha_k & (\text{if } \mathrm{in}(f, i) \in \mathcal{S}^{\mathbf{C}}) \end{cases}$$

and let

$$[t]^\alpha = \operatorname*{lub}_{k \in \mathbf{N}} [t]^\alpha_k$$

3. For a limit ordinal α, let $[-]^\alpha = \bigcup_{\iota < \alpha} [-]^\iota$. Note that $[t]^\iota = [t]^\kappa$ if $\iota < \kappa < \alpha$ and every $\lceil t \rceil < \iota$.

The *(infinitary) algebraic interpretation* derived from $\langle A, [-] \rangle$ is given as $[-]^{\boldsymbol{\Omega}} : \mathcal{G}^\star \to A$.

With a *variable interpretation* $\zeta : \mathcal{X} \to A$, we extend the domain of algebraic interpretation from $\mathcal{G}^\star$ to $\mathcal{T}^\star$; let $[x]^\xi = \xi(x)$ and similar as above for the other forms of a term. ◿

Remark 69 Our notation in the above definition was chosen for notational economy. Some explanation may be in order: we wish to require continuity of the constructor interpretations only with respect to their coinductive arguments, as explained above for the successor function (in a context where this is a inductive constructor). We do not mind the possible lack

of continuity in the inductive arguments, because they are subject to the properness constraint, and thus cannot be infinitely nested. For example, suppose f is a ternary inductive constructor with the first two arguments coinductive and the third inductive. Then we require that the functions $f(-,-,b)$ are continuous. This can neatly be expressed by the requirement

$$\mathrm{lub}\{[f](b_1,b_2,b_3) \mid b_i \in B_i\} = [f](\mathrm{lub}\, B_1, \mathrm{lub}\, B_2, \mathrm{lub}\, B_3)$$

together with the stipulation that $\mathrm{lub}\, B_3 = \mathrm{lub}\{b_3\} = b_3$ so that the above equation is actually equivalent to

$$\mathrm{lub}\{[f](b_1,b_2,b_3) \mid b_1 \in B_1,\ b_2 \in B_2\} = [f](\mathrm{lub}\, B_1, \mathrm{lub}\, B_2, b_3)$$

□

Remark 70 Definition 68 will be used in Definition 91 to ensure the well-definedness of the ordinal interpretation introduced there as a criterion for the DN property. The previous remark may be helpful in order to see that the two definitions actually match. Note in advance that in Definition 91 the continuity in the coinductive arguments is indeed ensured. On the coinductive argument places the constructor interpretation is there just the identity. For the inductive positions, the interpretation function is the successor, hence discontinuous, which does not harm by the properness constraint. □

2.6 Semantics

This section considers semantics of terms, which one might think of as a sorted version of the algebraic interpretation as introduced in the previous section. However, as we already indicated there, there is a contrast. We have introduced algebraic interpretations mainly for constructing ordinal assignments on terms which decrease under reduction, e.g. inductive height estimation (Definition 91). Switching to semantics, our primary concern is invariance of interpretations under conversion—and hence reduction. An order on the domain is no longer required.

Now a new issue arises. Invariance of an algebraic interpretation under finite reduction does not necessarily guarantee invariance under infinite reduction. We need again to make a requirement of continuity. This is illustrated in the following example, using the single rewrite rule

$$\mathsf{falses} \to \mathsf{F} : \mathsf{falses}$$

with the defined symbol $\mathsf{falses} : () \to \mathsf{STREAM}_{\mathsf{BOOL}}$ and the semantics (as a

sorted algebraic interpretation) given by

$$\begin{aligned}
\mathcal{A}_{\mathsf{BOOL}} &= \{\text{True}, \text{False}\} \\
\mathcal{A}_{\mathsf{STREAM}_{\mathsf{BOOL}}} &= \{\text{ExistsTrue}, \text{AllFalse}\} \\
[\![\mathsf{T}]\!] &= \text{True} \\
[\![\mathsf{F}]\!] &= \text{False} \\
[\![\mathsf{cons}^{\mathsf{BOOL}}]\!](a, b) &= \begin{cases} \text{ExistsTrue} & (\text{if } a = \text{True or } b = \text{ExistsTrue}) \\ \text{AllFalse} & (\text{if } a = \text{False and } b = \text{AllFalse}) \end{cases} \\
[\![\mathsf{falses}]\!] &= \text{ExistsTrue}
\end{aligned}$$

with ExistsTrue $\succ$ AllFalse. Then, since

$$[\![\mathsf{falses}]\!] = \text{ExistsTrue} = [\![\mathsf{F} : \mathsf{falses}]\!]$$

we have $\forall t, s.\ t \to s \Rightarrow [\![t]\!] = [\![s]\!]$, while $\mathsf{falses} \twoheadrightarrow \mathsf{F} : \mathsf{F} : \mathsf{F} : \ldots$ and

$$[\![\mathsf{F} : \mathsf{F} : \mathsf{F} : \ldots]\!] = \text{AllFalse} \neq [\![\mathsf{falses}]\!]$$

As a semantical interpretation, we wish that

$$[\![t]\!] = [\![\text{cnf}(t)]\!]$$

and, moreover, we wish to ensure the above condition as easily as in finitary TRS:

Desire 71 Given a semantical interpretation $[\![-]\!] : \mathcal{G}^\star \to \mathcal{A}$ in a productive algorithmic system, if

$$\forall t, s \in \mathcal{G}^\star.\ ((t \to s) \Rightarrow [\![t]\!] = [\![s]\!])$$

viz. the interpretation is preserved under single-step reduction, then the semantics would be adequate, i.e. $[\![t]\!] = [\![\text{cnf}(t)]\!]$ for all proper ground terms $t \in \mathcal{G}^\star$. ⊟

We will formalize the scheme of semantical interpretation on a productive algorithmic system with some restriction so that the above desire will be fulfilled.

2.6.1 Formalization

First, we introduce a metric on the proper terms.

Definition 72 Given a proper term t and a position $p \in \text{dom}(t)$, the *coinductive depth* of p in t, written $\|p\|_t$ is inductively given by

$$\begin{aligned}
\|\epsilon\|_t &= 0 \\
\|p \cdot n\|_t &= \|p\|_t + \begin{cases} 1 & (\text{if } t(p) \in \mathcal{C} \text{ and } \text{in}(t(p), n) \in \mathcal{S}^{\mathbf{C}}) \\ 0 & (\text{otherwise}) \end{cases}
\end{aligned}$$

◪

Definition 73 The metric function $d_{\mathcal{T}^\star} : \mathcal{T}^\star \times \mathcal{T}^\star \to [0,1)$ is defined by

$$d_{\mathcal{T}^\star}(t,s) = \begin{cases} 0 & (\text{if } t \equiv s) \\ 2^{-k} & (\text{otherwise}) \end{cases}$$

where $k = \min\{\|p\|_t \mid p \in \mathrm{dom}(t) \cap \mathrm{dom}(s) \text{ and } t(p) \neq s(p)\}$. ◿

Next, we introduce the notion of L-contraction functions:

Definition 74 Let $0 \leqslant L < 1$. Then, given metric spaces $A_1, \ldots, A_n$, and B, a function $f : A_1 \times \cdots \times A_n \to B$ is an L-*contraction* if

$$d_B(f(a_1, \ldots, a_n), f(a'_1, \ldots, a'_n)) \leqslant L \max\{d_{A_i}(a_i, a'_i) \mid i = 1, \ldots, n\}$$ ◿

Now, we formalize our scheme for semantical interpretation.

Definition 75 Given an algorithmic system, a *(denotational) semantics* $\mathcal{A}$ of the system consists of:

1. A set $\mathcal{A}_S$ for each sort $S \in \mathcal{S}$, as the *(denotational) domain* of S. For a coinductive sort $S \in \mathcal{S}^{\mathbf{C}}$, the domain $\mathcal{A}_S$ should be a complete metric space. Moreover, there should exist a fixed M such that

   $$d_S(a,b) \leqslant M$$

   for all $a, b \in \mathcal{A}_S$, for every $S \in \mathcal{S}^{\mathbf{C}}$. We write $\mathcal{A}$ to denote $\biguplus_{S \in \mathcal{S}} \mathcal{A}_S$.

2. For each function symbol $f \in \Sigma$ of type

   $$f : S_1 \times \cdots \times S_n \to T$$

   a *semantics function*

   $$f^{\mathcal{A}} : \mathcal{A}_{S_1} \times \cdots \times \mathcal{A}_{S_n} \to \mathcal{A}_T$$

   There should exist a fixed $L \in [0,1)$ such that $f^{\mathcal{A}}$ is L-contraction if $\mathrm{out}(f) \in \mathcal{S}^{\mathbf{C}}$, where we set

   $$d_{\mathcal{A}_S}(a,b) = \begin{cases} 0 & (\text{if } a = b) \\ M/L & (\text{if } a \neq b) \end{cases}$$

   as we are concerned with continuity only on the coinductive arguments. ◿

We write $\langle M, L \rangle$-semantics to specify M and L in the above restriction.

A semantics given within the above restriction induces the homomorphic interpretation $\mathcal{G}^\star \to \mathcal{A}$ with metric completion, similarly as in algebraic interpretation. However, for semantical interpretation, we do not assume the existence of a minimal value of the domain. Thus, we have to specify an element to take place of the minimal value for each coinductive sort. It will turn out that the induced interpretation does not depend on this choice.

Definition 76 Given an $\langle M, L\rangle$-semantics $\langle \mathcal{A}, (-)^{\mathcal{A}}\rangle$, let Θ be a function $\mathcal{S}^{\mathbf{C}} \to \mathcal{A}$ such that $\Theta(S) \in \mathcal{A}_S$ for each $S \in \mathcal{S}^{\mathbf{C}}$. Then, the *semantical interpretation* $[\![-]\!]^\Theta : \mathcal{G}^\star \to \mathcal{A}$ is defined as follows.

1. Let $\alpha \leqslant \mathbf{\Omega}$. Define $[\![-]\!]^\Theta_\alpha : \mathcal{G}^\star_\alpha \to \mathcal{A}$ by transfinite induction on α as follows, where $\mathcal{G}^\star_\alpha = \{t \in \mathcal{G}^\star \mid \lceil t\rceil < \alpha\}$.
   (a) When $\alpha = 0$, the domain $\mathcal{G}^\star_0$ is empty and thus $[\![-]\!]^\Theta_\alpha$ is the empty function.
   (b) For $\alpha > 0$, we assume $[\![-]\!]^\Theta_\beta$ is already defined for $\beta < \alpha$. Let $k \in \mathbf{N}$. Define $[\![-]\!]^\Theta_{\alpha;n} : \mathcal{G}^\star_\alpha \to \mathcal{A}$ inductively as

$$[\![f(t_1,\dots,t_n)]\!]^\Theta_{\alpha;k} = \begin{cases} \Theta(\mathrm{out}(f)) & (\text{if } f \in \mathcal{C},\ \mathrm{out}(f) \in \mathcal{S}^{\mathbf{C}},\ k = 0) \\ f^{\mathcal{A}}([\![t_1]\!]^\Theta_{\alpha;k-1},\dots,[\![t_n]\!]^\Theta_{\alpha;k-1}) & (\text{if } f \in \mathcal{C},\ \mathrm{out}(f) \in \mathcal{S}^{\mathbf{C}},\ k > 0) \\ f^{\mathcal{A}}([\![t_1]\!]^\Theta_{\lceil t_1\rceil},\dots,[\![t_n]\!]^\Theta_{\lceil t_n\rceil}) & (\text{otherwise}) \end{cases}$$

   Note that $\lceil t_i\rceil < \alpha$ for $i = 1,\dots,n$ in the last case.
   (c) We claim that $\langle [\![t]\!]^\Theta_{\alpha,k}\rangle_{k\in\mathbf{N}}$ forms a Cauchy sequence for every $t \in \mathcal{G}^\star$.

   Proof of the claim: Let $t \in \mathcal{G}^\star$ and $S = \mathrm{srt}(t)$. We show

$$d_{\mathcal{A}_S}([\![t]\!]^\Theta_{\alpha;k}, [\![t]\!]^\Theta_{\alpha;k+1}) \leqslant M L^k$$

   by mathematical induction on k, which will suffice. We write d to denote the lefthand-side of this inequation.
   For the base case $k = 0$, the inequality is trivial by the definition of M. For the induction step, let $t \equiv f(t_1,\dots,t_n)$. If $f \in \mathcal{D}$ or $S \in \mathcal{S}^{\mathbf{I}}$, then the value $[\![t]\!]^\Theta_{\alpha;k}$ does not depend on k, and thus we have $d = 0 \leqslant M L^k$. If $f \in \mathcal{C}$ and $S \in \mathcal{S}^{\mathbf{I}}$, then, since $f^{\mathcal{A}}$ is an L-contraction, with the induction hypothesis we have

$$\begin{aligned} d &\leqslant L \max\{d_{\mathcal{A}_{\mathrm{in}(f,i)}}([\![t_i]\!]^\Theta_{\alpha;k-1}, [\![t_i]\!]^\Theta_{\alpha;k}) \mid i = 1,\dots,n\} \\ &\leqslant L(M L^{k-1}) = M L^k \end{aligned}$$

   ⌟

(d) Let $[\![t]\!]^{\Theta}_{\alpha} = \lim_{k\to\infty}[\![t]\!]^{\Theta}_{\alpha;k}$. Note that every Cauchy sequence in the complete space has its limit.

2. Let $[\![t]\!]^{\Theta} = [\![t]\!]^{\Theta}_{\boldsymbol{\Omega}}$. ◿

As mentioned above, we do not have to specify a function Θ:

Lemma 77 *Let $\langle \mathcal{A}, (-)^{\mathcal{A}}\rangle$ be a semantics. Then, the interpretations $[\![-]\!]^{\Theta}$ and $[\![-]\!]^{\Theta'}$ agree on each other for all applicable Θ and Θ'.*

Proof: Let the semantics be an $\langle M, L\rangle$ semantics. Fix an applicable pair of $\Theta, \Theta' : \mathcal{S}^{\mathbf{C}} \to \mathcal{A}$, and an arbitrary positive real number ε. We show that

$$\mathrm{d}_{\mathcal{A}_{\mathrm{srt}(t)}}([\![t]\!]^{\Theta}, [\![t]\!]^{\Theta'}) < \varepsilon$$

which will suffice.

Since $\langle [\![t]\!]^{\Theta}_k\rangle_{k\in\mathbf{N}}$ and $\langle [\![t]\!]^{\Theta'}_k\rangle_{k\in\mathbf{N}}$, as in Definition 76 are both Cauchy sequences, respectively leading to $[\![t]\!]^{\Theta}$ and $[\![t]\!]^{\Theta'}$, we can find $k_0 \in \mathbf{N}$ such that $k \geqslant k_0$ implies

$$\mathrm{d}_{\mathcal{A}_{\mathrm{srt}(t)}}([\![t]\!]^{\Theta}, [\![t]\!]^{\Theta}_k), \mathrm{d}_{\mathcal{A}_{\mathrm{srt}(t)}}([\![t]\!]^{\Theta'}, [\![t]\!]^{\Theta'}_k) < \varepsilon/3$$

On the other hand, we can find $k_1 \in \mathbf{N}$ such that $M\,L^{k_1} < \varepsilon/3$. Similarly as the claim in Definition 76 (1c), we have

$$\mathrm{d}_{\mathcal{A}_{\mathrm{srt}(t)}}([\![t]\!]^{\Theta}_k, [\![t]\!]^{\Theta'}_k) \leqslant M\,L^k$$

Now, let $k_2 = k_0 \vee k_1$. Then,

$$\begin{aligned}
&\mathrm{d}_{\mathcal{A}_{\mathrm{srt}(t)}}([\![t]\!]^{\Theta}, [\![t]\!]^{\Theta'}) \\
&\leqslant \mathrm{d}_{\mathcal{A}_{\mathrm{srt}(t)}}([\![t]\!]^{\Theta}, [\![t]\!]^{\Theta}_{k_2}) + \mathrm{d}_{\mathcal{A}_{\mathrm{srt}(t)}}([\![t]\!]^{\Theta}_{k_2}, [\![t]\!]^{\Theta'}_{k_2}) + \mathrm{d}_{\mathcal{A}_{\mathrm{srt}(t)}}([\![t]\!]^{\Theta'}_{k_2}, [\![t]\!]^{\Theta'}) \\
&< \varepsilon/3 + \varepsilon/3 + \varepsilon/3 = \varepsilon
\end{aligned}$$

□

So, we write $[\![-]\!]$ to denote the semantical interpretation induced from the semantics $\langle \mathcal{A}, (-)^{\mathcal{A}}\rangle$.

Lemma 78 *Let $[\![-]\!] : \mathcal{G}^{\star} \to \mathcal{A}$ be a semantics. Then, for any $\varepsilon > 0$, there exists δ such that $t, s \in \mathcal{G}^{\star}$ and $\mathrm{d}_{\mathcal{T}^{\star}}(t, s) < \delta$ implies $\mathrm{d}_{\mathcal{A}_{\mathrm{srt}(t)}}([\![t]\!], [\![s]\!]) < \varepsilon$.*

Proof: Similarly shown as Lemma 77. □

Corollary 79 *If an infinite sequence of proper ground terms $t_0, t_1, t_2, \ldots$ converges to a proper ground term s, then*

$$[\![s]\!] = \lim_{n\to\infty}[\![t_n]\!]$$

□

2.6.2 Adequacy

Definition 80 Given a semantical interpretation $[\![-]\!] : \mathcal{G}^\star \to \mathcal{A}$ on a productive system, the semantics is *adequate* if $[\![t]\!] = [\![\mathrm{cnf}(t)]\!]$ for all $t \in \mathcal{G}^\star$. ◲

Lemma 81 *Let $[\![-]\!] : \mathcal{G}^\star \to \mathcal{A}$ be a semantical interpretation on a productive algorithmic system. Then the following conditions are all equivalent:*

1. *For every $t, s \in \mathcal{G}^\star$, $t \to s$ implies $[\![t]\!] = [\![s]\!]$, viz. semantics is preserved under single-step reduction.*
2. *For every $t, s \in \mathcal{G}^\star$, $t \twoheadrightarrow s$ implies $[\![t]\!] = [\![s]\!]$, viz. semantics is preserved under infinite reduction.*
3. *The semantics is adequate.*

Proof: (1⇒2) Follows from Corollary 79, using Compression Lemma (Lemma 38).
(2⇒3) Since $t \twoheadrightarrow \mathrm{cnf}(t)$, we have $[\![t]\!] = [\![\mathrm{cnf}(t)]\!]$.
(3⇒1) Since $t \to s$ implies $\mathrm{cnf}(t) \equiv \mathrm{cnf}(s)$, We have

$$[\![t]\!] = [\![\mathrm{cnf}(t)]\!] = [\![\mathrm{cnf}(s)]\!] = [\![s]\!]$$

□

Combining this result with the following definition, we can say: given a 'full and weakly sound' semantics, a productive system specifies the function $[\![f]\!]$ as $f^{\mathcal{A}}$ for each defined function symbol f (c.f. Figure 7).

Definition 82 Let a semantical interpretation $[\![-]\!] : \mathcal{V}^\star \to \mathcal{A}$ (that can be regarded as a semantics under assumption $\mathcal{D} = \varnothing$) be given. Then:

1. The semantics is *full* if for every $a \in \mathcal{A}$ there exists some $t \in \mathcal{V}^\star$ such that $[\![t]\!] = a$.
2. The semantics is *strongly sound* if, for every $t, s \in \mathcal{V}^\star$, $[\![t]\!] = [\![s]\!]$ implies $t \equiv s$.
3. The semantics is *weakly sound* if, for every n-ary defined symbol f and every $t_1, \ldots, t_n, s_1, \ldots, s_n \in \mathcal{V}^\star$, $[\![t_i]\!] = [\![s_i]\!]$ for $i = 1, \ldots n$ implies $[\![\mathrm{cnf}(f(t_1, \ldots, t_n))]\!] = [\![\mathrm{cnf}(f(s_1, \ldots, s_n))]\!]$. ◲

Proposition 83 *A strongly sound semantics is always weakly sound.* □

Lemma 84 *Let a semantical interpretation $[\![-]\!] : \mathcal{V}^\star \to \mathcal{A}$ be given. Then:*

1. *If the semantics is full, then there exists at most one adequate semantical interpretation $[\![-]\!]' : \mathcal{G}^\star \to \mathcal{A}$ which extends $[\![-]\!] : \mathcal{V}^\star \to \mathcal{A}$.*

2. *If the semantics is weakly sound, then there exists some adequate semantical interpretations* $[\![-]\!] : \mathcal{G}^\star \to \mathcal{A}$ *which extend* $[\![-]\!] : \mathcal{V}^\star \to \mathcal{A}$.

Proof:

1. Let $[\![-]\!]_1, [\![-]\!]_2 : \mathcal{G}^\star \to \mathcal{A}$ be adequate semantical interpretations satisfying $[\![t]\!]_1 = [\![t]\!]_2 = [\![t]\!]$ for all $t \in \mathcal{V}^\star$.

   For any n-ary defined function symbol f and $a_1 \in \mathcal{A}_{\mathrm{in}(f,1)}$, $\ldots$, $a_n \in \mathcal{A}_{\mathrm{in}(f,n)}$, since the semantics $[\![-]\!]$ is full, we can find some proper constructor normal forms $t_1, \ldots, t_n \in \mathcal{V}^\star$ such that $[\![t_i]\!] = a_i$. From adequacy of the semantics, we have

$$\begin{aligned} f^{\mathcal{A}}(a_1, \ldots, a_n) &= [\![f(t_1, \ldots, t_n)]\!] \\ &= [\![\mathrm{cnf}(f(t_1, \ldots, t_n))]\!] \end{aligned}$$

   Thus, $f^{\mathcal{A}}$ is unique if it exists and induces an adequate semantics.

2. Follows from the definition of weak soundness.

□

2.6.3 Examples

We present the standard semantics for the sorts and the constructors presented in Subsection 2.1.1.

Natural numbers

For the sort NAT and the constructors $0 : () \to \mathsf{NAT}$ and $s : \mathsf{NAT} \to \mathsf{NAT}$, let

$$\begin{aligned} \mathcal{A}_{\mathsf{NAT}} &= \mathbf{N} \\ 0^{\mathcal{A}}() &= 0 \\ s^{\mathcal{A}}(n) &= n + 1 \end{aligned}$$

This semantics is full and strongly sound.

Lists

Given a sort S with $\mathcal{A}_S$, the sort LIST_S and the constructors $[]^S : () \to \mathsf{LIST}_S$ and $\mathsf{cons}^S_{\mathsf{fin}} : S \times \mathsf{LIST}_S \to \mathsf{LIST}_S$, let

$$\begin{aligned} \mathcal{A}_{\mathsf{LIST}_S} &= \mathcal{A}_S^* \\ {[]^S}^{\mathcal{A}}() &= \epsilon \\ {\mathsf{cons}^S_{\mathsf{fin}}}^{\mathcal{A}}(a, l) &= a; l \end{aligned}$$

where we overload the symbol ';' to denote the construction of a list. This semantics is full and strongly sound.

Streams

Also for streams, we overload the symbol ':' for stream construction. We write $\mathbf{a}$ to denote $a_0 : a_1 : a_2 : \ldots$.

Given an inductive sort S with $\mathcal{A}_S$, we let

$$\mathcal{A}_{\mathsf{STREAM}_S} = (\mathcal{A}_S)^\omega$$

with

$$d_{(\mathcal{A}_S)^\omega}(\mathbf{a}, \mathbf{b}) = \sup\{d_{\mathcal{A}_S}(a_k, b_k)/2^{k+1} \mid k \in \mathbf{N}\}$$

Note that $d_{(\mathcal{A}_S)^\omega}(\mathbf{a}, \mathbf{b}) \leqslant 1/2$ for any pair of streams $\mathbf{a}$ and $\mathbf{b}$.

Now, let

$$\mathsf{cons}^{S\,\mathcal{A}}(a, l) = a : l$$

Observe that $\mathsf{cons}^{S\,\mathcal{A}}$ is $\frac{1}{2}$-continuous.

This semantics is full or weakly/strongly sound if the semantics on the sort S is so, respectively.

Tree ordinals

For the sort ORD and the constructors $\mathsf{O} : () \to \mathsf{ORD}$, $\mathsf{S} : \mathsf{ORD} \to \mathsf{ORD}$, and $\mathsf{L} : \mathsf{STREAM}_{\mathsf{ORD}} \to \mathsf{ORD}$, let

$$\begin{aligned} \mathcal{A}_{\mathsf{ORD}} &= \mathbf{\Omega} \\ \mathsf{O}^{\mathcal{A}}() &= 0 \\ \mathsf{S}^{\mathcal{A}}(\alpha) &= \alpha + 1 \\ \mathsf{L}^{\mathcal{A}}(\langle \alpha_i \rangle_{i \in \mathbf{N}}) &= \sup_{i \in \mathbf{N}} \alpha_i \end{aligned}$$

This semantics is full but not strongly sound (because of $\mathsf{L}^{\mathcal{A}}$. For a counterexample, consider

$$0 = [\![\mathsf{O}]\!] = [\![\mathsf{O} : \mathsf{O} : \mathsf{O} : \ldots]\!]$$

However, the systems on tree ordinals presented in this thesis are all weakly sound.

As to a counterexample of a not weakly sound system, put the defined function $\mathsf{hd} : \mathsf{STREAM}_{\mathsf{ORD}} \to \mathsf{ORD}$ with the rule

$$\mathsf{hd}(n : x) \to n$$

Observe that this symbol disturbs weak soundness of the system.

Chapter 3

Ensuring productivity

In this thesis, we are concerned with productivity of algorithmic systems. As stated as Corollary 40, productivity of the algorithmic system is equivalent to the conjunction of WN, DN, and CN. Conversely, there are three essential kinds of failure of productivity of algorithmic systems: ¬WN, ¬DN, and ¬CN. Here we present a counterexample for each kind of the failure:

1. (not WN) The defined symbol $\mathsf{rmv0} : \mathsf{STREAM_{NAT}} \to \mathsf{STREAM_{NAT}}$ with the rules

$$\begin{aligned} \mathsf{rmv0}(0 : x) &\to \mathsf{rmv0}(x) \\ \mathsf{rmv0}(\mathsf{s}(n) : x) &\to x \end{aligned}$$

   removing the leading zeros from the given stream. This system is DN and CN, but not WN since the term $\mathsf{rmv0}(0 : 0 : 0 : \dots)$ has no infinitary normal form.

2. (not DN) The defined symbol $\mathsf{cnt0} : \mathsf{STREAM_{NAT}} \to \mathsf{NAT}$ with the rules

$$\begin{aligned} \mathsf{cnt0}(0 : x) &\to \mathsf{s}(\mathsf{cnt0}(x)) \\ \mathsf{cnt0}(\mathsf{s}(n) : x) &\to 0 \end{aligned}$$

   counting the number of leading zeros of the given stream. This system is WN and CN, but not DN, since the term $\mathsf{cnt0}(0 : 0 : 0 : \dots)$ has the constructor normal form $\mathsf{s}\,\mathsf{s}\,\mathsf{s}\dots$, which is not proper.

3. (not CN) Consider the system with the defined symbols

$$\begin{aligned} \mathsf{sum} &: \mathsf{STREAM_{NAT}} \to \mathsf{NAT} \\ + &: \ \mathsf{NAT} \times \mathsf{NAT} \to \mathsf{NAT} \end{aligned}$$

   with the rule

$$\begin{aligned} \mathsf{sum}(n : y) &\to n + \mathsf{sum}(y) \\ n + 0 &\to n \\ n + (\mathsf{s}\ m) &\to \mathsf{s}(n + m) \end{aligned}$$

   This system is WN and DN, but not CN.

So, in order to analyze productivity, we study the properties WN, DN, and CN. We devote a separate section to each of these three properties. We begin with the DN property, since the WN section will use the result of the DN section.

3.1 Domain normalization

In this section we study the domain normalization (DN) property. We recall the definition of DN:

Definition A semantically sorted system is DN if no initially proper ground term contains a constructor vicious path, that is, an infinite nesting of inductive constructors. ◹

First, we characterize the DN property by observing the construction process of a constructor vicious path. Second, based on that characterization, we give a criterion for DN by means of an algebraic interpretation. Thus, finding a certain algebraic interpretation will suffice to establish DN.

3.1.1 Characterization via path generation

To characterize the DN property of an algorithmic system, we first consider the case where the system is not DN. That is, when the system admits a constructor vicious path in an initially proper ground term s. From the definition of initial properness, we can find a proper ground term t such that $t \to^{\omega} s$ by Compression Lemma (Lemma 38). Note that t is proper and thus contains no vicious path. Since properness of the term is preserved under finite reduction, the constructor vicious path in s is generated exactly along the infinite reduction. We are interested in the process, or mechanism, of this generation.

First, we formalize the generation of paths.

Definition 85 A *path generation sequence* is an infinite sequence of terms, concatenated with $\searrow$ (see Definition 61) and $\twoheadrightarrow$, containing infinitely many $\searrow$. For example,

$$
\begin{aligned}
\mathsf{omega} &\twoheadrightarrow \mathsf{L}(\mathsf{O} : \mathsf{S}\,\mathsf{O} : \mathsf{nats}(\mathsf{S}\,\mathsf{S}\,\mathsf{O})) \\
&\quad \searrow \mathsf{O} : \mathsf{S}\,\mathsf{O} : \mathsf{nats}(\mathsf{S}\,\mathsf{S}\,\mathsf{O}) \\
&\quad \searrow \mathsf{S}\,\mathsf{O} : \mathsf{nats}(\mathsf{S}\,\mathsf{S}\,\mathsf{O}) \\
&\quad \twoheadrightarrow \mathsf{S}\,\mathsf{O} : \mathsf{S}\,\mathsf{S}\,\mathsf{O} : \mathsf{nats}(\mathsf{S}\,\mathsf{S}\,\mathsf{S}\,\mathsf{O}) \\
&\quad \searrow \mathsf{S}\,\mathsf{S}\,\mathsf{O} : \mathsf{nats}(\mathsf{S}\,\mathsf{S}\,\mathsf{S}\,\mathsf{O}) \\
&\quad \searrow \mathsf{nats}(\mathsf{S}\,\mathsf{S}\,\mathsf{S}\,\mathsf{O}) \\
&\quad \ldots
\end{aligned}
$$

The *trace* of a path generation sequence is the concatenation of the root symbol of the term immediately before each $\searrow$. For the above example, the trace is

$$\mathsf{L} \cdot \mathsf{cons}^{\mathsf{ORD}} \cdot \mathsf{cons}^{\mathsf{ORD}} \cdot \mathsf{cons}^{\mathsf{ORD}} \cdot \ldots$$

◪

Definition 86 Given an infinite path $\langle n_i \rangle_{i \in \mathbf{N}}$ in a term t, the *trace of the path* is given as $t(\epsilon) \cdot t(n_0) \cdot t(n_0 \cdot n_1) \cdot \ldots \cdot t(n_0 \cdot \ldots \cdot n_k) \cdot \ldots$. ◪

Lemma 87 *A path in an infinitary reduct of a term coincides with a path generation sequence with the same initial term, i.e.*

1. *If* $t \twoheadrightarrow\!\!\!\!\!\rightarrow s$ *and* p *is an infinite path in* s*, then there exists a path generation sequence whose trace agrees on the trace of* p*.*
2. *Given a path generation sequence beginning with a term* t*, there exists a term* s *containing the trace of the path generation sequence as a path, such that* $t \rightarrow^{\omega} s$*.*

Proof: Both the claims easily follow from the definition of strong convergence of the reduction sequence. □

Corollary 88 *If* $t \twoheadrightarrow\!\!\!\!\!\rightarrow s$ *and* s *contains a constructor vicious path, then there exists a path generation sequence from* t *containing no* $\searrow^{\mathcal{D}}$ *but infinitely many* $\searrow^{\mathbf{I}}$*.* □

Next, we characterize the DN property as follows.

Theorem 89 *An algorithmic system is* DN *if there exists a* $\searrow^{\mathbf{I}} | \searrow^{\mathbf{C}} \rightarrow$*-compatible well-founded quasiorder on* $\mathcal{G}^{\star}$*.*

Proof: Let $\preccurlyeq$ be a $\searrow^{\mathbf{I}} | \searrow^{\mathbf{C}} \rightarrow$-compatible well-founded quasiorder on $\mathcal{G}^{\star}$. Let $t \in \mathcal{G}^{\star\twoheadrightarrow\!\!\!\!\!\rightarrow}$. Then, for every path p in t, by Lemma 87, we can find a path generation sequence beginning with a proper term whose trace agrees on the trace of p. Thus, if the trace of p contains no defined function symbol, then the trace of the generation sequence consists of $\searrow^{\mathbf{I}}$, $\searrow^{\mathbf{C}}$, and $\twoheadrightarrow$. From well-foundedness and $\searrow^{\mathbf{I}} | \searrow^{\mathbf{C}} \rightarrow$-compatibility of $\preccurlyeq$, the generation sequence can contain only finitely many $\searrow^{\mathbf{I}}$. Hence, t can contain no constructor vicious path. □

It should be remarked that the above theorem holds also for semantically sorted systems.

Theorem 90 *If an algorithmic term rewriting system is* DN*, then there exists a* $\searrow^{\mathbf{I}} | \searrow^{\mathbf{C}} \rightarrow$*-compatible well-founded quasiorder on* $\mathcal{G}^{\star}$*.*

Proof: Let $\mathcal{G}_\bot$ be the set of (ground) terms generated with the symbols $\mathcal{C} \uplus \{\bot_S \mid S \in \mathcal{S}\}$, where each $\bot_S$ is a fresh nullary constructor symbol with $\mathrm{out}(\bot_S) = S$. Let $\mathcal{G}^\star_\bot$ be the set of proper terms possibly containing bottoms. Then, for every proper term $t \in \mathcal{G}^\star_\bot$, we define the *quasi-constructor-normal-form* of t, written $\mathrm{qcnf}(t)$, coinductively as follows.

- If there exists a $s \in \mathcal{T}$ such that $t \twoheadrightarrow s$ and $s(\epsilon) \in \mathcal{C}$, then let
$$\mathrm{qcnf}(t) \equiv (s(\epsilon))(\mathrm{qcnf}(s/1), \ldots, \mathrm{qcnf}(s/\mathrm{ar}(s(\epsilon))))$$
- Otherwise, let $\mathrm{qcnf}(t) \equiv \bot$.

Note that $\mathrm{qcnf}(t) \in \mathcal{G}^\star_\bot$ for every $t \in \mathcal{G}^\star$, provided that the system is DN. Readers who are familiar with the lambda calculus [2] might regard quasi-constructor-normal-forms as analogous to Böhm trees.

Now, let $t \preccurlyeq s$ iff $\lceil \mathrm{qcnf}(t) \rceil \leqslant \lceil \mathrm{qcnf}(s) \rceil$. Then:

1. If $t \mathrel{\downarrow^{\mathbf{I}}} s$, then we have $t(\epsilon) \in \mathcal{C}$ and thus $\mathrm{qcnf}(t) \mathrel{\downarrow^{\mathbf{I}}} \mathrm{qcnf}(s)$ from functionality of the algorithmic system. By Proposition 63, we have $\lceil \mathrm{qcnf}(t) \rceil > \lceil \mathrm{qcnf}(s) \rceil$.
2. If $t \mathrel{\downarrow^{\mathbf{C}}} s$, then we have $\mathrm{qcnf}(t) \mathrel{\downarrow^{\mathbf{C}}} \mathrm{qcnf}(s)$ and thus we have $\lceil \mathrm{qcnf}(t) \rceil \geqslant \lceil \mathrm{qcnf}(s) \rceil$ similarly as above.
3. if $t \to s$, then we have $\mathrm{qcnf}(t) \equiv \mathrm{qcnf}(s)$ from the above definition of quasi-constructor-normal-forms. Thus, $\lceil \mathrm{qcnf}(t) \rceil = \lceil \mathrm{qcnf}(s) \rceil$.

So, $\preccurlyeq$ is $\mathrel{\downarrow^{\mathbf{I}}} \mid \mathrel{\downarrow^{\mathbf{C}}} \to$-compatible. Since $\mathbf{\Omega}$ is well-founded, so is $\preccurlyeq$. □

3.1.2 Inductive height estimation

The characterization of DN given in the previous section is so general that it can be used to ensure the DN property of *any* DN algorithmic system, even if the system is not productive. However, for practical applications, i.e. proving DN for concrete examples, this characterization is only of limited value, due to its generality. Therefore, we now give a more concrete criterion for DN that we can actually use. This criterion will be the existence of a certain algebraic interpretation.

Definition 91 An *inductive height estimator* $\mathbf{H}$ consists of a monotonic function $H^f : \mathbf{\Omega}^n \to \mathbf{\Omega}$ for each n-ary defined function symbol f. Given an inductive height estimator $\mathbf{H}$, the *inductive height estimation* is an algebraic interpretation (Section 2.5) given by the continuous Σ-algebra $\langle \mathbf{\Omega}, [\![-]\!]_{\mathbf{H}} \rangle$ defined by

$$[\![f]\!]_{\mathbf{H}}(\alpha_1, \ldots, \alpha_n) = \begin{cases} \max\{\alpha_i + \chi^{\mathbf{I}}_{\mathrm{in}}(f, i) \mid i = 1, \ldots, n\} & (\text{if } f \in \mathcal{C}) \\ H^f(\alpha_1, \ldots, \alpha_n) & (\text{if } f \in \mathcal{D}) \end{cases}$$

An inductive height estimator is *adequate* if the corresponding inductive height estimation $[\![-]\!]^{\zeta}_{\mathbf{H}} : \mathcal{T}^\star \to \mathbf{\Omega}$ satisfies $[\![l]\!]^{\zeta}_{\mathbf{H}} \geqslant [\![r]\!]^{\zeta}_{\mathbf{H}}$ for every rule $(l, r) \in \mathcal{R}$ and every variable interpretation $\zeta : \mathcal{X} \to \mathbf{\Omega}$. ◪

A comment on the use of the notion of algebraic interpretation in this definition has already been given in Remark 70.

Theorem 92 *An algorithmic system is* DN *if there exists an adequate inductive height estimator.*

Proof: Let $[\![-]\!]^{\zeta}_{\mathbf{H}} : \mathcal{T}^\star \to \mathbf{\Omega}$ be the inductive height estimation derived from an adequate inductive height estimator. Define the order $\preccurlyeq$ on $\mathcal{G}^\star$ as $t \preccurlyeq s$ iff $[\![t]\!]_{\mathbf{H}} \leqslant [\![s]\!]_{\mathbf{H}}$. Since $\langle \mathbf{\Omega}, \leqslant \rangle$ is well-ordered, so is $\langle \mathcal{G}^\star, \preccurlyeq \rangle$.

Suppose $t, s \in \mathcal{G}^\star$ and $t \to_p s$. Then, we can find some $(l, r) \in \mathcal{R}$ and $\sigma : \mathcal{X} \to \mathcal{G}^\star$ such that $t/p \equiv l\sigma$ and $s \equiv t\{p \mapsto r\sigma\}$. By adequacy of the estimation, we have $l\sigma \succcurlyeq r\sigma$. From monotonicity of the estimation, we have $t \succcurlyeq s$. Thus, $\preccurlyeq$ is $\varnothing \,|\, \to$-compatible. In addition, by the definition of inductive height estimation, $\preccurlyeq$ is $\searrow^{\mathbf{I}} | \searrow^{\mathbf{C}}$-compatible.

Hence, $\preccurlyeq$ forms a $\searrow^{\mathbf{I}} | \searrow^{\mathbf{C}} \!\!\to$-compatible well-founded quasiorder on $\mathcal{G}^\star$. By Theorem 89, the system is DN. □

Remark 93 The reverse implication of the above theorem does not hold; there exists some productive algorithmic system which admits *no* adequate inductive height estimator. Consider the system

$$\begin{aligned} \mathsf{f}(\mathsf{T}) &\to 0 \\ \mathsf{f}(\mathsf{F}) &\to \mathsf{s}(\mathsf{f}(\mathsf{T})) \end{aligned}$$

For any inductive height estimator $\mathbf{H}$, we have

$$[\![\mathsf{f}(\mathsf{F})]\!]_{\mathbf{H}} = \mathsf{H}^{\mathsf{f}}([\![\mathsf{F}]\!]_{\mathbf{H}}) = \mathsf{H}^{\mathsf{f}}(0)$$

and

$$[\![\mathsf{s}(\mathsf{f}(\mathsf{T}))]\!]_{\mathbf{H}} = [\![\mathsf{f}(\mathsf{T})]\!]_{\mathbf{H}} + 1 = \mathsf{H}^{\mathsf{f}}(0) + 1$$

Thus, we cannot have $[\![\mathsf{f}(\mathsf{F})]\!]_{\mathbf{H}} \geqslant [\![\mathsf{s}(\mathsf{f}(\mathsf{T}))]\!]_{\mathbf{H}}$. This counterexample is from Jörg Endrullis [15].

To overcome this counterexample, one may generalize inductive height estimation to give an interpretation not only to defined symbols, but also to constructor symbols. However, for future work, we aim to automate detection of an adequate inductive height estimator for some productive algorithmic systems. ⧈

Now, we demonstrate how we can easily find an adequate inductive height estimator to ensure DN. The following examples show that the systems HAM and ORD-AME are both DN.

Example 94 We show that the system HAM is DN. Let $\mathbf{H}$ be an inductive height estimator, and let $\zeta : \mathcal{X} \to \mathbf{\Omega}$ be a variable interpretation. Here we write $[x]$ to denote $\zeta(x)$ for better readability. For $\mathbf{H}$ to be adequate, it is required that $[\![l]\!]^{\zeta}_{\mathbf{H}} \geqslant [\![r]\!]^{\zeta}_{\mathbf{H}}$ for all $(l, r) \in \mathcal{R}$. According to this requirement we translate the rewrite rules as in Table 14 into inequations. For example, the inequalities

$$\begin{aligned} H^{+}([n], 0) &\geqslant [n] \\ H^{+}([n], [m] + 1) &\geqslant H^{+}([n], [m]) + 1 \\ H^{\times}([n], 0) &\geqslant 0 \\ H^{\times}([n], [m] + 1) &\geqslant H^{+}(H^{\times}([n], [m]), [n]) \end{aligned}$$

are required for H^{+} and $H^{\times}$. Thus, we set

$$\begin{aligned} H^{+}(\alpha, \beta) &= \alpha + \beta \\ H^{\times}(\alpha, \beta) &= \alpha \cdot \beta \end{aligned}$$

Similarly, we set

$$\begin{aligned} H^{\mathsf{cmp}}(\alpha, \beta) &= 0 \\ H^{\mathsf{mrg}}(\beta, \gamma) &= \sup\{\beta, \gamma\} \\ H^{\mathsf{aux}}(\alpha, \beta, \gamma) &= \sup\{\beta, \gamma\} \end{aligned}$$

It remains to define H^{sca}, H^{Ham}, H^{Ham2}, H^{Ham3}, and H^{Ham5}. Since Ham, Ham2, Ham3, and Ham5 are all nullary and supposed to compute some unbounded streams of natural numbers, we let

$$H^{\mathsf{Ham}} = H^{\mathsf{Ham2}} = H^{\mathsf{Ham3}} = H^{\mathsf{Ham5}} = \omega$$

The required inequality on Ham is then

$$\omega \geqslant H^{\mathsf{sca}}(\omega, 2), H^{\mathsf{sca}}(\omega, 3), H^{\mathsf{sca}}(\omega, 5)$$

which makes it non-trivial to define H^{sca}. Because of the rule

$$\mathsf{sca}(n : x, m) \to (n \times m) : \mathsf{sca}(x, m)$$

we must have

$$H^{\mathsf{sca}}(\sup\{[n] + 1, [x]\}, [m]\}) \geqslant \sup\{[n] \times [m] + 1, H^{\mathsf{sca}}([x], [m])$$

As long as H^{sca} is weakly increasing,

$$H^{\mathsf{sca}}(\sup\{[n] + 1, [x]\}, [m]\}) \geqslant H^{\mathsf{sca}}([x], [m])$$

always holds. So, the requirement can be simplified as

$$\mathsf{H}^{\mathsf{sca}}([\mathfrak{n}]+1,[\mathfrak{m}]) \geqslant [\mathfrak{n}] \times [\mathfrak{m}] + 1$$

Of course, letting $\mathsf{H}^{\mathsf{sca}}(\alpha,\beta) = \alpha \times \beta + 1$ obviously satisfies the above inequation and monotonicity. However, if we defined $\mathsf{H}^{\mathsf{sca}}$ so, then we would have $\mathsf{H}^{\mathsf{sca}}(\omega,2) = \omega \times 2 + 1 > \omega$, disturbing the required inequality on $\mathsf{H}^{\mathsf{Ham}}$. In order to fix the problem, we let

$$\mathsf{H}^{\mathsf{sca}}(\alpha,\beta) = \sup\{\iota \cdot \beta + 1 \mid \iota < \alpha\}$$

Note that $\mathsf{H}^{\mathsf{sca}}(\omega,\mathfrak{n}) = \omega$ for every $\mathfrak{n} \in \mathbf{N}$.

Then, $\mathbf{H}$ defined as above forms an adequate inductive height estimator. The system HAM is hence DN. ⊡

Example 95 We show that the system ORD-AME as shown in Table 15 is DN. Let

$$\begin{aligned}
\mathsf{H}^{+}(\alpha,\beta) = \mathsf{H}^{\mathsf{add}_{\mathsf{L}}}(\alpha,\beta) &= \alpha+\beta+1\\
\mathsf{H}^{\cdot}(\alpha,\beta) = \mathsf{H}^{\mathsf{mul}_{\mathsf{L}}}(\alpha,\beta) &= (\alpha+1)\cdot(\beta+1)\\
\mathsf{H}^{\mathsf{exp}}(\alpha,\beta) = \mathsf{H}^{\mathsf{exp}_{\mathsf{L}}}(\alpha,\beta) &= (2\cdot(\alpha+1))^{\beta+1}\\
\mathsf{H}^{\mathsf{pow0}_{\mathsf{L}}}(\alpha) &= 2\cdot(\alpha+1)\\
\mathsf{H}^{\mathsf{omega}} &= \omega\\
\mathsf{H}^{\mathsf{nats}}(\alpha) &= \alpha+\omega
\end{aligned}$$

Observe that the derived inductive estimation satisfies $[\![\mathfrak{l}]\!]^{\zeta}_{\mathbf{H}} \geqslant [\![\mathfrak{r}]\!]^{\zeta}_{\mathbf{H}}$ for every $(\mathfrak{l},\mathfrak{r}) \in \mathcal{R}$ and every $\zeta : \mathcal{X} \to \mathbf{\Omega}$. Hence, the system is DN.

The trickiest point is to find $\mathsf{H}^{\mathsf{exp}}(\alpha,\beta) = (2\cdot(\alpha+1))^{\beta+1}$ satisfying $[\![\mathsf{exp}(\mathfrak{n},\mathsf{S}\,\mathfrak{m})]\!]^{\zeta}_{\mathbf{H}} \geqslant [\![\mathsf{exp}(\mathfrak{n},\mathfrak{m})\cdot\mathfrak{n}]\!]^{\zeta}_{\mathbf{H}}$;

$$\begin{aligned}
[\![\mathsf{exp}(\mathfrak{n},\mathsf{S}\,\mathfrak{m})]\!]^{\zeta}_{\mathbf{H}} &= (2\cdot([\mathfrak{n}]+1))^{[\mathfrak{m}]+2}\\
&= (2\cdot([\mathfrak{n}]+1))^{[\mathfrak{m}]+1}\cdot 2\cdot([\mathfrak{n}]+1)\\
&= ((2\cdot([\mathfrak{n}]+1))^{[\mathfrak{m}]+1} + (2\cdot([\mathfrak{n}]+1))^{[\mathfrak{m}]+1})\cdot([\mathfrak{n}]+1)\\
&\geqslant ((2\cdot([\mathfrak{n}]+1))^{[\mathfrak{m}]+1}+1)\cdot([\mathfrak{n}]+1)\\
&= [\![\mathsf{exp}(\mathfrak{n},\mathfrak{m})\cdot\mathfrak{n}]\!]^{\zeta}_{\mathbf{H}}
\end{aligned}$$

where $[\mathfrak{n}]$ and $[\mathfrak{m}]$ denote $\zeta(\mathfrak{n})$ and $\zeta(\mathfrak{m})$, respectively.

In addition, it should be noticed that all the components $\langle \mathsf{H}^{\mathsf{f}}\rangle_{\mathsf{f}\in\mathcal{D}}$ of the above inductive height estimator are not only monotonic but also strongly increasing. This fact will be employed in Example 107. ⊡

3.2 Infinitary normalization

This section studies the WN property. As a consequence of Lemma 51 and Corollary 53, the following four infinitary normalization properties of an algorithmic system are all equivalent.

1. Weak normalization with respect to proper ground terms (WN).
2. Strong normalization with respect to proper ground terms.
3. Weak normalization with respect to proper terms.
4. Strong normalization with respect to proper terms.

Therefore, we will refer to the property simply as 'infinitary normalization', as this section is named. Moreover, Corollary 52 gives a rephrasing of the property:

> An algorithmic system is WN if there exists no infinite root-active reduction sequence beginning with a proper term.

In this section we will use this characterization of WN; we characterize the WN property by the non-existence of infinite root-active reduction sequence, and an infinite root-active reduction sequence can be projected to an infinite descending sequence. Infinitary normalization is thus characterized by a certain quasiorder on proper ground terms. In addition, we will give a criterion for WN by estimating the number of possible root-reduction steps.

3.2.1 Characterization by quasiorder

As discussed in Section 2.3, Infinitary normalization of an algorithmic system can be characterized as follows.

Theorem 96 *An algorithmic system is* WN *if there exists a* $\to_\epsilon \mid \to$*-compatible well-founded quasiorder on* $\mathcal{G}^\star$*.*

Proof: Suppose that $\preccurlyeq$ is a $\to_\epsilon \mid \to$-compatible well-founded quasiorder on $\mathcal{G}^\star$. Let

$$t_0 \to t_1 \to t_2 \to \dots$$

be an infinite reduction sequence of proper ground terms. Then, we have

$$t_0 \succcurlyeq t_1 \succcurlyeq t_2 \succcurlyeq \dots$$

from $\varnothing \mid \to$-compatibility of $\preccurlyeq$. Moreover, from $\to_\epsilon \mid \varnothing$-compatibility of $\preccurlyeq$, we have $t_n \succ t_{n+1}$ whenever $t_n \to_\epsilon t_{n+1}$. Thus, by well-foundedness of $\preccurlyeq$, there can be only finitely many root-reduction steps. Hence, the system is WN. □

It should be remarked that the above theorem holds also for semantically sorted systems.

Theorem 97 *If an algorithmic system is* WN*, then the system admits an* $\to_\epsilon \mid \to$*-compatible well-founded quasiorder on* $\mathcal{G}^\star$*.*

Proof: Suppose that the system is WN. By Lemmas 39 and 51, and Corollary 53, for every proper ground term t, we have the unique infinitary normal form $\mathrm{nf}(t)$.

Let $\mathrm{ra}(t)$ be defined as follows. Roughly speaking, $\mathrm{ra}(t)$ indicates the *root activity*; the number of root-reduction steps to reach the infinitary normal form from t.

1. Let $t_0 \equiv t$.
2. Suppose that t_n is already given. Then, replace all the immediate subterms of t_n by their normal form, and let t'_n be the result of this replacement. Formally, let
$$t'_n = (t(\epsilon))(t_n^1, \ldots, t_n^{\mathrm{ar}(t(\epsilon))})$$
where $t_n^i = \mathrm{nf}(t_n/i)$ for $i = 1, \ldots, \mathrm{ar}(t(\epsilon))$. Note that $t_n \twoheadrightarrow_{\neq\epsilon} t'_n$.
3. Suppose that t'_n is already given, and that $t'_n/i \in \mathcal{N}$ for all applicable i. If $t'_n \in \mathcal{N}$, then let $\mathrm{ra}(t) = n$. Otherwise, since every t'_n/i is already in normal form, the redex of t'_n must occur at the root. By algorithmicity of the system, the redex is unique. Let t_{n+1} be the result of root-reduction step from t'_n. Note that $t'_n \to_\epsilon t_{n+1}$.

Since $t_n \twoheadrightarrow t'_n \to_\epsilon t_{n+1}$ for every n, we will eventually reach the normal form; otherwise an infinite root-active reduction sequence would be produced, contradicting the WN property.

Now, let $t \preccurlyeq s$ iff $\mathrm{ra}(t) \leqslant \mathrm{ra}(s)$. By definition of ra, the partial order $\preccurlyeq$ is $\to_\epsilon \mid \to$-compatible. Since $\mathbf{N}$ is well-ordered, $\preccurlyeq$ is well-founded. □

Remark 98 In the above proof, $\mathrm{ra}(t)$ is actually the minimal number of the root-reduction steps to reach the infinitary normal form of t. This fact is proved in [32]. ⧈

3.2.2 Root activity estimation

In the previous section we have given a criterion for the DN property as an algebraic interpretation, which estimates (or approximates) the inductive height $\lceil \mathrm{cnf}(t) \rceil$ from above, for every proper term t. Likewise, one might think of ensuring the WN property by estimating $\mathrm{ra}(t)$ as in the proof of Theorem 97 for every proper term t, by means of an algebraic interpretation.

However, a naive attempt to estimate $\mathrm{ra}(t)$ by an algebraic interpretation will not succeed. For a counterexample, consider the addition function

$$\begin{aligned} n + 0 &\to n \\ n + s\, m &\to s(n + m) \end{aligned}$$

Assume that a function $R^+ : \mathbf{N} \to \mathbf{N}$ satisfies $\mathrm{ra}(n+m) \leqslant R^+(\mathrm{ra}(n), \mathrm{ra}(m))$. Then, for any $i \in \mathbf{N}$, we have $\mathrm{ra}(O + \ulcorner i \urcorner) \leqslant R^+(\mathrm{ra}(O), \mathrm{ra}(\ulcorner i \urcorner)) = R^+(0, 0)$. However, obviously we have $\mathrm{ra}(O + \ulcorner i \urcorner) = i$ from the definition of root activity. Letting $i = R^+(0, 0) + 1$ leads to contradiction.

In fact, $\mathrm{ra}(n + m)$ depends on $\lceil \mathrm{cnf}(m) \rceil$, rather than $\mathrm{ra}(n)$ or $\mathrm{ra}(m)$. Thus, in this section we suppose that we have already found an adequate inductive height estimator $\mathbf{H}$, and formalize the notion of root activity estimation as follows.

Definition 99 Let $[\![-]\!]^{\zeta}_{\mathbf{H}} : \mathcal{T}^\star \to \mathbf{\Omega}$ be an adequate inductive height estimation. Then, a *root activity estimator* $\mathbf{R}$ consists of a monotonic function $R^f : \mathbf{\Omega}^n \to \mathbf{\Omega}$ for each n-ary defined function symbol f. Given a root activity estimator $\mathbf{R}$ and a variable interpretation $\zeta : \mathcal{X} \to \mathbf{\Omega}$, the *root activity estimation* is defined by

$$\begin{aligned} (\!|f(t_1, \ldots, t_n)|\!)^{\zeta}_{\mathbf{R}} &= \begin{cases} 0 & (\text{if } f \in \mathcal{C}) \\ R^f([\![t_1]\!]^{\zeta}_{\mathbf{H}}, \ldots, [\![t_n]\!]^{\zeta}_{\mathbf{H}}) & (\text{if } f \in \mathcal{D}) \end{cases} \\ (\!|x|\!)^{\zeta}_{\mathbf{R}} &= \zeta(x) \end{aligned}$$

A root activity estimator is *adequate* if the corresponding root activity estimation $(\!|-|\!)^{\zeta}_{\mathbf{R}} : \mathcal{T}^\star \to \mathbf{\Omega}$ satisfies $(\!|l|\!)^{\zeta}_{\mathbf{R}} > (\!|r|\!)^{\zeta}_{\mathbf{R}}$ for every rule $(l, r) \in \mathcal{R}$ and every variable interpretation $\zeta : \mathcal{X} \to \mathbf{\Omega}$. ◪

Theorem 100 *An algorithmic system is* WN *if there exists an adequate root activity estimator.*

Proof: Let $\mathbf{H}$ be an adequate inductive height estimator, and $\mathbf{R}$ be an adequate root activity estimator. Define the order $\preccurlyeq$ on $\mathcal{G}^\star$ as $t \preccurlyeq s$ iff $(\!|t|\!)_{\mathbf{R}} \leqslant (\!|s|\!)_{\mathbf{R}}$. Since $\langle \mathbf{\Omega}, \leqslant \rangle$ is well-ordered, so is $\langle \mathcal{G}^\star, \preccurlyeq \rangle$.

Suppose $t, s \in \mathcal{G}^\star$ and $t \to_p s$. Then, we can find some $(l, r) \in \mathcal{R}$ and $\sigma : \mathcal{X} \to \mathcal{G}^\star$ such that $t/p \equiv l\sigma$ and $s \equiv t\{p \mapsto r\sigma\}$. If $p = \epsilon$, then we have $t \equiv l\sigma$ and $s \equiv r\sigma$. From adequacy of the root activity estimator, $t \succ s$ follows. If $p \neq \epsilon$, then we have $t/p \succcurlyeq s/p$ from adequacy of the inductive height estimator, and thus $t \succcurlyeq s$ from monotonicity of the root activity estimator. Thus, $\preccurlyeq$ forms a $\varnothing \,|\, \to$-compatible well-founded quasiorder on $\mathcal{G}^\star$. Hence, by Theorem 96, the system is WN. □

Example 101 We show that the system HAM is WN. Recall the adequate inductive height estimation in Example 94:

$$\begin{aligned}
H^{+}(\alpha,\beta) &= \alpha+\beta\\
H^{\times}(\alpha,\beta) &= \alpha\cdot\beta\\
H^{\mathsf{sca}}(\alpha,\beta) &= \sup\{\iota\cdot\beta+1 \mid \iota<\alpha\}\\
H^{\mathsf{cmp}}(\alpha,\beta) &= 0\\
H^{\mathsf{mrg}}(\alpha,\beta) &= \sup\{\alpha,\beta\}\\
H^{\mathsf{aux}}(\alpha,\beta,\gamma) &= \sup\{\beta,\gamma\}\\
H^{\mathsf{Ham}} = H^{\mathsf{Ham2}} &= H^{\mathsf{Ham3}} = H^{\mathsf{Ham5}} = \omega
\end{aligned}$$

Let $\mathbf{R}$ be a root activity estimator, and ζ be a variable interpretation. Again we write $[x]$ for $\zeta(x)$. For $\mathbf{R}$ to be adequate, we compute $(\!|l|\!)^{\zeta}_{\mathbf{R}}$ and $(\!|r|\!)^{\zeta}_{\mathbf{R}}$ for every $(l,r)\in\mathcal{R}$, to obtain the required inequalities:

$$\begin{aligned}
R^{+}([n],0) &> [n]\\
R^{+}([n],[m]+1) &> 0\\
R^{\times}([n],0) &> 0\\
R^{\times}([n],[m]+1) &> R^{+}([n]\cdot[m],[n])\\
R^{\mathsf{sca}}([n],\sup\{[m]+1,[x]\}) &> 0\\
R^{\mathsf{cmp}}(0,0) &> 0\\
R^{\mathsf{cmp}}([n]+1,0) &> 0\\
R^{\mathsf{cmp}}(0,[m]+1) &> 0\\
R^{\mathsf{cmp}}([n]+1,[m]+1) &> R^{\mathsf{cmp}}([n],[m])\\
R^{\mathsf{mrg}}([nx],[my]) &> R^{\mathsf{aux}}(0,[nx],[my])\\
R^{\mathsf{aux}}(0,[nx],[my]) &> 0\\
R^{\mathsf{aux}}(0,[x],[my]) &> 0\\
R^{\mathsf{aux}}(0,[nx],[y]) &> 0\\
R^{\mathsf{Ham}} &> 0\\
R^{\mathsf{Ham2}} &> R^{\mathsf{sca}}(\omega,2)\\
R^{\mathsf{Ham3}} &> R^{\mathsf{sca}}(\omega,3)\\
R^{\mathsf{Ham5}} &> R^{\mathsf{sca}}(\omega,5)
\end{aligned}$$

where $[nx]=\sup\{[n]+1,[x]\}$ and $[my]=\sup\{[m]+1,[y]\}$. Note that $(\!|t|\!)^{\zeta}_{\mathbf{R}}=0$ whenever $t(\epsilon)\in\mathcal{C}$. Thus, an adequate root activity estimator is easier to find than an adequate inductive height estimator. Let

$$\begin{aligned}
R^{+}(\alpha,\beta) &= \alpha+1\\
R^{\times}(\alpha,\beta) &= \alpha\cdot\beta+2
\end{aligned}$$

$$\begin{aligned} \mathsf{R}^{\mathsf{sca}}(\alpha,\beta) &= 1 \\ \mathsf{R}^{\mathsf{cmp}}(\alpha,\beta) &= \sup\{\alpha,\beta\}+1 \\ \mathsf{R}^{\mathsf{mrg}}(\alpha,\beta) &= 2 \\ \mathsf{R}^{\mathsf{aux}}(\alpha,\beta,\gamma) &= 1 \\ \mathsf{R}^{\mathsf{Ham}} &= 1 \\ \mathsf{R}^{\mathsf{Ham2}} = \mathsf{R}^{\mathsf{Ham3}} = \mathsf{R}^{\mathsf{Ham5}} &= 2 \end{aligned}$$

and observe that **R** is adequate. The system is thus WN. ⊡

It is an easy exercise to show that also the system ORD-AME is WN.

Remark 102 One might expect the opposite implication; if the algorithmic system is DN and WN, and an adequate inductive height estimator **H** is given, is there always an adequate root activity estimator? We will see a counterexample in Chapter 4 (Remark 165). ⧈

3.3 Constructor normalization

This section studies the constructor normalization (CN) property. We recall the second version of the definition of CN (Definition 30.5):

Definition A semantically sorted infinitary rewriting system is CN if no initially proper ground normal form contains a functional vicious path, i.e. a path containing infinitely many defined symbols. ◪

In particular for algorithmic systems, we can rephrase the definition (c.f. Lemma 33):

Definition An algorithmic term rewriting system is CN if no initially proper ground normal form contains a defined function symbol. ◪

First, we will attempt to characterize the CN property by a well-founded quasiorder on $\mathcal{G}^\star$, as we have done for DN and WN in the preceding two sections. However, the attempt will fail to characterize CN but give a characterization of SCN (strong constructor normalization, Definition 30.6):

Definition A semantically sorted infinitary rewriting system is SCN if no initially proper ground term contains a functional vicious path. ◪

The gap between CN and SCN is that SCN disallows any initially proper ground term to contain a functional vicious path, while CN allows it as long as the term is not in a normal form. It is a little difficult to know if an initially proper ground term is a normal form in the framework of a compatible well-founded quasiorder.

So, in order to characterize CN, we will introduce another notion than well-founded quasiorder, which we shall call 'observation'. Roughly speaking, to approximate a function application $f(t_1, \ldots, t_n)$, we have to approximate the arguments $t_1, \ldots, t_n$. The notion of observation formalizes quantities of those approximations.

3.3.1 Strong constructor normalization

From Lemma 87, the next lemma follows analogously as Corollary 88.

Lemma 103 *If* $t \twoheadrightarrow s$ *and* s *contains a functional vicious path, then there exists a path generation sequence from* t *containing infinitely many* $\searrow^{\mathcal{D}}$. □

The following corollary is immediate.

Corollary 104 *An algorithmic term rewriting system is* CN *if there exists some* $\searrow^{\mathcal{D}} | \searrow\!\!\rightarrow$*-compatible well-founded quasiorder on* $\mathcal{G}^\star$. □

Remark 105 The reverse implication of the above corollary fails in general; a productive algorithmic term rewriting system does not always admits a $\searrow^{\mathcal{D}} | \searrow\!\!\rightarrow$-compatible well-founded quasiorder on $\mathcal{G}^\star$. For a counterexample, consider the system HAM. Assume that $\preccurlyeq$ be a $\searrow^{\mathcal{D}} | \searrow\!\!\rightarrow$-compatible well-founded quasiorder on $\mathcal{G}^\star$. Then, from

$$\begin{aligned}\mathsf{Ham} &\to \mathsf{s}\,0 : \mathsf{mrg}(\mathsf{mrg}(\mathsf{Ham2}, \mathsf{Ham3}), \mathsf{Ham5})\\ &\searrow^{\mathbf{C}} \mathsf{mrg}(\mathsf{mrg}(\mathsf{Ham2}, \mathsf{Ham3}), \mathsf{Ham5})\\ &\searrow^{\mathcal{D}} \mathsf{Ham5}\\ &\to \mathsf{sca}(\mathsf{Ham}, \mathsf{s}\,\mathsf{s}\,\mathsf{s}\,\mathsf{s}\,\mathsf{s}\,0)\\ &\searrow^{\mathcal{D}} \mathsf{Ham}\end{aligned}$$

we have $\mathsf{Ham} \succ \mathsf{Ham}$, contradicting to irreflexivity of $\prec$.

The essence of this counterexample is as follows: consider the system

$$\begin{aligned}\mathsf{id}(n) &\to n\\ \mathsf{zeros} &\to 0 : \mathsf{id}(\mathsf{zeros})\end{aligned}$$

The system is obviously productive but admits no $\searrow^{\mathcal{D}} | \searrow\!\!\rightarrow$-compatible well-founded quasiorder; $\mathsf{zeros} \to 0 : \mathsf{id}(\mathsf{zeros}) \searrow^{\mathbf{C}} \mathsf{id}(\mathsf{zeros}) \searrow^{\mathcal{D}} \mathsf{zeros}$. In point of fact, a recursive specification of a nullary symbol nested by another (already-)defined function symbol disables such a quasiorder, even if the occurrence of another defined symbol is 'guarded' by a coinductive constructor. ⧈

As in the above remark, a $\searrow^{\mathcal{D}} | \searrow\!\!\rightarrow$-compatible well-founded quasiorder on $\mathcal{G}^\star$ prohibits any generation of a functional vicious path. Actually, such a quasiorder does not characterize CN but SCN.

Theorem 106 *An algorithmic term rewriting system is* SCN *if and only if there exists a* $\searrow^{\mathcal{D}} | \searrow\!\!\rightarrow$*-compatible well-founded quasiorder on* $\mathcal{G}^\star$*.*

Proof: (If) Trivial.

(Only If) For each $\alpha \in \mathbf{\Omega}$, define the subset $\mathcal{G}^{\mathcal{D}}_\alpha \subseteq \mathcal{G}^\star$ by transfinite induction on α as $t \in \mathcal{G}^{\mathcal{D}}_\alpha$ iff for any s such that $t \twoheadrightarrow s$ and $p \in \mathrm{dom}(s)$, if there exists some $q \lhd p$ such that $s(q) \in \mathcal{D}$, then there exists some $\beta < \alpha$ such that $s/p \in \mathcal{G}^{\mathcal{D}}_\beta$. Let $\mathcal{G}^{\mathcal{D}}_{\mathbf{\Omega}} = \bigcup_{\alpha\in\mathbf{\Omega}} \mathcal{G}^{\mathcal{D}}_\alpha$. We claim that $\mathcal{G}^{\mathcal{D}}_{\mathbf{\Omega}} = \mathcal{G}^\star$.

To proof the claim by contradiction, assume $t \in \mathcal{G}^\star \setminus \mathcal{G}^{\mathcal{D}}_{\mathbf{\Omega}}$. Let the set X consist of pairs $(s,p) \in \mathcal{G}^\star \times \mathbf{N}_+^*$ such that $t \twoheadrightarrow s$, $p \in \mathrm{dom}(s)$, and that $\exists q \lhd p.\ s(q) \in \mathcal{D}$. Note that X is a countable set. If there exists some $\beta^s_p \in \mathbf{\Omega}$ for every $(s,p) \in X$ satisfying $(s,p) \in \mathcal{G}^{\mathcal{D}}_{\beta^s_p}$, then we have $t \in \mathcal{G}^{\mathcal{D}}_\alpha$ where $\alpha = \sup\{\beta^s_p + 1 \mid (s,p) \in X\}$, contradicting that $t \notin \mathcal{G}^{\mathcal{D}}_{\mathbf{\Omega}}$. Thus, we can find some $(s,p) \in X$ such that $s/p \in \mathcal{G}^\star \setminus \mathcal{G}^{\mathcal{D}}_{\mathbf{\Omega}}$. By iterating this argument with letting $t := s/p$, we obtain a convergent infinite reduction generating a vicious path with infinitely many defined symbols nested, which contradicts the SCN property of the system. Hence, $\mathcal{G}^{\mathcal{D}}_{\mathbf{\Omega}} = \mathcal{G}^\star$.

Now, define the function $D : \mathcal{G}^\star \to \mathbf{\Omega}$ by $D(t) = \min\{\alpha \in \mathbf{\Omega} \mid t \in \mathcal{G}^{\mathcal{D}}_\alpha\}$. Clearly, D induces a $\searrow^{\mathcal{D}} | \searrow\!\!\rightarrow$-compatible well-founded quasiorder on $\mathcal{G}^\star$. □

Example 107 We show that the system ORD-AME is strongly productive. Recall the inductive height estimator as in Example 95, which in fact was even strongly monotonic. Thus, the partial order on $\mathcal{G}$ derived from the corresponding inductive height estimation is $\searrow^{\mathcal{D}}\searrow^{\mathbf{I}} | \searrow^{\mathbf{C}}\!\!\rightarrow$-compatible and thus $\searrow^{\mathcal{D}} | \searrow\!\!\rightarrow$-compatible. Therefore, by Theorem 106, the system is SCN. Since it has been already shown that the system is DN and WN, by Corollary 41, the system ORD-AME is strongly productive. ⊡

3.3.2 Characterization by observation

In order to characterize the CN property, one may think of finding some relations R and R' on $\mathcal{G}^\star$ or $\mathcal{T}^\star$ such that the existence of a $R | R'$-compatible well-founded quasiorder is sufficient and necessary for CN. However, that will turn out not so effective. Consider the following two one-rule systems

$$f(n : m : x) \to n : m : f(f(x)) \tag{A}$$

and

$$f(n : m : x) \to n : f(f(x)) \tag{B}$$

Observe that the system (A) is CN (and also productive) while the system (B) is not. In each of these systems, the computation of $f(n : m : x)$ depends

on that of $f(f(x))$ with

$$\begin{aligned} f(n:m:x) &\to n:m:f(f(x)) \\ &\searrow^{C} m:f(f(x)) \\ &\searrow^{C} f(f(x)) && \text{(for (A))} \\ f(n:m:x) &\to n:f(f(x)) \\ &\searrow^{C} f(f(x)) && \text{(for (B))} \end{aligned}$$

Thus, to require a relation between $f(n:m:x)$ and $f(f(x))$, or between every pair of instances of those terms, will not help establishing the CN property; it cannot distinguish a CN system (A) from a non-CN system (B). Informally speaking, the property WN, DN, or SCN says that something should not occur for infinitely many times. So, we were able to characterize those properties by well-founded quasiorders. On the other hand, the CN property mentions normal forms. Therefore, a characterization of CN will have a different flavour.

In the present work, we follow the quantitative analysis presented in [18], which has originated from [44] analyzing productivity of stream specifications. The notion of gauges generalizes the framework of quantitative analysis of streams to that of infinite terms.

Henceforth, we assume that the system is algorithmic, DN, and WN. We will call such a system *'a quasi-productive algorithmic system'*. Under this assumption, the CN property is equivalent to productivity. The remainder of this section will study productivity of a given quasi-productive algorithmic system.

Definition 108 A defined function symbol f is *productive* if every proper ground term of the form $f(t_1,\ldots,t_n)$ where $t_i \in \mathcal{V}^\star$ for every applicable i has a constructor normal form. ◿

Lemma 109 *A quasi-productive algorithmic system is productive if and only if every defined function symbol is productive.*

Proof: (If) Suppose that the system is quasi-productive and that every defined function symbol is productive. Let t be a proper ground term. Without loss of generality, we can assume $t(\epsilon) \in \mathcal{D}$ (otherwise, we can make a parallel computation on all the non-nested occurrence of defined function symbols).

Let $T_\alpha = \{t \in \mathcal{G}^\star \mid t(\epsilon) \in \mathcal{D} \text{ and } \lceil t \rceil = \alpha\}$ for every $\alpha \in \mathbf{\Omega}$. To show that $t \in T_\alpha$ implies that t has a constructor normal form, for every $\alpha \in \mathbf{\Omega}$, will suffice. We show that by transfinite induction on α.

For the base case $\alpha = 0$, we have $t \equiv f$ for some $f \in \mathcal{D}$ whenever $t \in T_0$. The claim follows from productivity of f.

For the transfinite induction step, assume that every term in $\bigcup_{\iota<\alpha} \mathsf{T}_\iota$ has a constructor normal form. Then, for every $t \equiv f(t_1, \ldots, t_n) \in \mathsf{T}_\alpha$, from the definition of inductive height, we have $\lceil t_i \rceil < \lceil t \rceil = \alpha$ for every $i = 1, \ldots, n$. Thus, by the induction hypothesis, each t_i has the constructor normal form t'_i. Hence, $t \twoheadrightarrow f(t'_1, \ldots, t'_n)$. Since the system is DN, each t'_i is proper. The claim now follows from productivity of f.

(Only If) Trivial. □

In order to characterize the productivity of defined function symbols, we formalize the notion of observation.

Definition 110 Let S be a sort.

- A *constructor pattern* of sort S is a finite term generated by the symbols $\mathcal{C} \cup \{\bot_S \mid S \in \mathcal{S}\}$. We write $\mathcal{P}^S$ to denote the set of constructor patterns of sort S. In the sequel we will refer to a 'constructor pattern' concisely as a 'pattern'.

- Define the set $\mathcal{T}_\bot$ to consist of the terms possibly containing some bottoms, i.e. the terms generated by the symbols $\Sigma \cup \{\bot_S \mid S \in \mathcal{S}\}$. Given terms $t, s \in \mathcal{T}_\bot$, we write $t \blacktriangleright s$ if s can be obtained by replacing some subterms t/p in t by $\bot_{\mathrm{out}(t(p))}$. Formally, $t \blacktriangleright s$ iff there exists $P \subseteq \mathrm{dom}(t)$ such that

$$s(p) = \begin{cases} t(p) & \text{(if there exists no } q \in P \text{ such that } q \trianglelefteq p) \\ \bot_{\mathrm{out}(t(p))} & \text{(if } p \in P) \\ \text{undefined.} & \text{(otherwise)} \end{cases}$$

  Given a pattern u, we write $\wr u \int$ to denote the set $\{t \in \mathcal{T}_\bot \mid t \blacktriangleright u\}$, viz. the *refinements* of the pattern u.

- An *observation* of sort S is a subset $\gamma \subseteq \mathcal{P}^S$ such that $\bigcup_{u\in\gamma} \wr u \int \supseteq \mathcal{V}^{\star S}$, viz. a set of patterns which cover all the proper constructor normal forms of sort S. The set of observations of sort S is denoted by $\mathcal{O}^S$. Given an observation γ, we write $\mathbb{l}\gamma\mathbb{J}$ to denote the set $\bigcup_{u\in\gamma} \wr u \int$. ◪

Lemma 111 *If* $t \blacktriangleright \cdot \twoheadrightarrow s$, *then* $t \twoheadrightarrow \cdot \blacktriangleright s$.

Proof: Suppose $t \blacktriangleright s_0 \to s_1 \to \ldots \to s_n \equiv s$. Then, by induction on n, we can construct terms $t_0, \ldots, t_n$ such that $t \equiv t_0 \to t_1 \to \ldots \to t_n \blacktriangleright s_n$ and $t_i \blacktriangleright t_n$ for every $i = 0, \ldots, n$ (see Figure 20). Note that concealing subterms by bottoms does not create a redex, because no rule contains $\bot$.

□

$$\begin{array}{ccccccccccc} t & \equiv & t_0 & \to & t_1 & \to & \cdots & \to & t_n & & \\ & & \blacktriangledown & & \blacktriangledown & & & & \blacktriangledown & & \\ & & s_0 & \to & s_1 & \to & \cdots & \to & s_n & \equiv & s \end{array}$$

Figure 20: *Lemma 111*

Lemma 112 *If* $t \twoheadrightarrow \cdot \blacktriangleright s$ *and* s *is finite, then there exists a finite* t' *such that* $t \blacktriangleright t' \twoheadrightarrow \cdot \blacktriangleright s$.

Proof: Suppose $t \twoheadrightarrow s' \blacktriangleright s$. We assume that $t \to s'$; the other case will follow by induction on the length of reduction steps. Let p be the position of the contracted redex in the step $t \to s'$, (l, r) be the applied rule, and σ be a substitution such that $s' \equiv t\{p \mapsto r\sigma\}$.

Let P_s be the positions in s' which remains in s, i.e.

$$P_s = \{q \in \mathrm{dom}(s) \mid s(q) \neq \bot\}$$

(note that $\mathrm{dom}(s') \supseteq \mathrm{dom}(s)$). Then, define the sets of positions Q_1, Q_2, and Q_3 by

$$\begin{aligned} Q_1 &= \{q \in P_s \mid p \not\trianglelefteq q\} \\ Q_2 &= \{p \cdot q \mid q \in \mathrm{dom}(l) \text{ and } l(q) \in \Sigma\} \\ Q_3 &= \left\{ p \cdot q_1 \cdot q_2 \;\middle|\; \begin{array}{l} q_1 \in \mathrm{dom}(l),\ l(q_1) \in \mathcal{X},\ q_2 \in \mathrm{dom}(\sigma(l(q_1))), \text{ and} \\ \exists q_3 \in r.\ r(q_3) = l(q_1),\ p \cdot q_3 \cdot q_2 \in P_s \end{array} \right\} \end{aligned}$$

Note that each of these sets is finite and a subset of $\mathrm{dom}(t)$. To give an intuition, Q_1 consists of the positions that remain in s and are out of the reduction; Q_2 consists of those forming the pattern of the contracted redex; Q_3 consists of those remaining in s, copied by the reduction. Thus, we can find a finite t' such that $t \blacktriangleright t'$, where all the positions in $Q_1 \cup Q_2 \cup Q_3$ remain. By the definition of Q_1, Q_2, and Q_3, we have $t' \to \cdot \blacktriangleright s$. □

Remark 113 This lemma is called 'prefix property' in [5]. ⧈

Lemma 114 *Let* S *be a sort. Then we have* $\bigcap_{\gamma \in \mathcal{O}^S} [\![\gamma]\!] = \mathcal{V}^{\star S}$.

Proof: Let S be a sort. Define the *truncation* of a term t at a depth n, written $\mathrm{trunc}(t, k)$, inductively by

$$\begin{aligned} \mathrm{trunc}(t, 0) &\equiv \bot_{\mathrm{srt}(t)} \\ \mathrm{trunc}(f(t_1, \ldots, t_n), k+1) &\equiv f(t'_1, \ldots, t'_n) \end{aligned}$$

where $t'_i \equiv \mathrm{trunc}(t_i, k)$ for every $i = 1, \ldots, n$. Now, let

$$\gamma_k = \{\mathrm{trunc}(t, k) \mid t \in \mathcal{V}^S\}$$

for every $k \in \mathbf{N}$. Observe that each γ_k forms an observation. Thus, we have

$$\bigcap_{\gamma \in \mathcal{O}^S} \llbracket \gamma \rrbracket \subseteq \bigcap_{k \in \mathbf{N}} \llbracket \gamma_k \rrbracket$$

Suppose $t \in \llbracket \gamma_k \rrbracket$. Then, for every $p \in \mathrm{dom}(t)$ such that $|p| < k$, we have $t(p) \in \mathcal{C}$, by definition of γ_k. Thus we have

$$\bigcap_{k \in \mathbf{N}} \llbracket \gamma_k \rrbracket \subseteq \mathcal{V}^S$$

Combine these set inclusion relations to obtain the claim. □

Example 115 The set $\mathcal{V}^{\star \mathsf{NAT}}$ itself forms an observation. Note that every proper constructor normal form of the sort NAT is finite. ⊡

We extend the notion of observation for patterns of multiple terms.

Definition 116 An *observation* of sort $S_1 \times \cdots \times S_n$ is a subset

$$\gamma \subseteq \mathcal{P}^{S_1} \times \cdots \times \mathcal{P}^{S_n}$$

such that $\llbracket \gamma \rrbracket \supseteq \mathcal{V}^{\star S_1} \times \cdots \times \mathcal{V}^{\star S_n}$, where

$$\llbracket \gamma \rrbracket = \{(t_1, \ldots, t_n) \mid \exists (u_1, \ldots, u_n) \in \gamma.\ t_i \in \lbrack u_i \rbrack \text{ for } i = 1, \ldots, n\}$$ ◪

Example 117 Given an algorithmic term rewriting system and $f \in \mathcal{D}$, let $\gamma = \{(l'/1, \ldots, l'/\mathrm{ar}(f)) \mid (l, r) \in \mathcal{R} \text{ and } l(\epsilon) = f\}$, where l' is the result of replacing every variable by $\bot$ of the same sort. Then, by case-exhaustivity of the algorithmic system, γ forms an observation of the input sort of f. ⊡

Remark 118 Given observations $\gamma_1 \in \mathcal{O}^{S_1}, \ldots, \gamma_n \in \mathcal{O}^{S_n}$, the cartesian product $\gamma_1 \times \cdots \times \gamma_n$ forms an observation of the sort $S_1 \times \cdots \times S_n$. However, the opposite implication does not hold; some observations of multiple sorts cannot be given in the form of a cartesian product of single-sort observations. We will see a counterexample in Example 121. ⧈

Definition 119 Let $f : S_1 \times \cdots \times S_n \to T$ be a defined function symbol. Then, an *observation requirement function* for f is a function $\rho : \mathcal{O}^T \to \mathcal{O}^{S_1 \times \cdots \times S_n}$ satisfying that, for every $\delta \in \gamma^T$ and every $(u_1, \ldots, u_n) \in \rho(\delta)$, there exists $v \in \delta$ such that $f(u_1, \ldots, u_n) \twoheadrightarrow \cdot \blacktriangleright v$. ◪

Theorem 120 *In a quasi-productive algorithmic system, a defined function symbol* f *is productive if and only if there exists an observation requirement function for* f.

Proof: (If) Let ρ be an observation requirement function for a defined function symbol f, of type $S_1 \times \cdots \times S_n \to T$. Define $\delta_k \in \mathcal{O}^T$ by

$$\delta_k = \{\mathrm{trunc}(t, k) \mid t \in \mathcal{V}^T\}$$

where trunc is as in the proof of Lemma 114. Suppose $t \equiv f(t_1, \ldots, t_n)$, where $t_i \in \mathcal{V}^\star$ for every $i = 1, \ldots, n$. Then, for every $k \in \mathbf{N}$, we can find patterns $(u_1, \ldots, u_n) \in \rho(\delta_k)$ and $v \in \delta_k$ such that

$$t \equiv f(t_1, \ldots, t_n) \blacktriangleright f(u_1, \ldots, u_n) \twoheadrightarrow \cdot \blacktriangleright v$$

By Lemma 111, we have $t \twoheadrightarrow \cdot \blacktriangleright v$, and thus we can find intermediate term $t_k \in \mathcal{G}^\star$ such that $t \twoheadrightarrow t_k \blacktriangleright v \in \delta_k$. Since $t_k \blacktriangleright v \in \delta_k$, from the definition of δ_k, t_k consists of only constructor symbols up to depth k. By finitary confluence of the algorithmic system, we have

$$t \twoheadrightarrow t_0 \twoheadrightarrow t_1 \twoheadrightarrow t_2 \twoheadrightarrow \ldots$$

Since a constructor symbol does not form a redex, the reduction sequence strongly converges to the constructor normal form of t. Since the system is quasi-productive, that constructor normal form is proper.

(Only If) Suppose that a defined function symbol $f : S_1 \times \cdots \times S_n \to T$ is productive, and let $\delta \in \mathcal{O}^T$ be an observation. Define $\rho(\delta)$ by

$$\rho(\delta) = \{u \in \mathcal{P}^{S_1 \times \cdots \times S_n} \mid f(S_1 \times \cdots \times S_n) \twoheadrightarrow \cdot \blacktriangleright {}^{\exists}v \in \delta\}$$

To show $\rho(\delta) \in \mathcal{O}^{S_1 \times \cdots \times S_n}$ will suffice. That is, by definition, for every $(t_1, \ldots, t_n) \in \mathcal{V}^{\star S_1} \times \cdots \times \mathcal{V}^{\star S_n}$, there should exist $(u_1, \ldots, u_n) \in \rho(\delta)$ such that $(t_1, \ldots, t_n) \blacktriangleright (u_1, \ldots, u_n)$. Suppose $(t_1, \ldots, t_n) \in \mathcal{V}^{\star S_1} \times \cdots \times \mathcal{V}^{\star S_n}$. Since f is productive, $f(t_1, \ldots, t_n)$ has a constructor normal form s. Since δ is an observation, we can find $v \in \delta$ such that $s \blacktriangleright v$. Since v is finite, by strong convergence, we can find s' such that $f(t_1, \ldots, t_n) \twoheadrightarrow s' \blacktriangleright v$. By Lemma 112, we can find a finite $t' \equiv f(t'_1, \ldots, t'_n)$ such that $t \blacktriangleright t'$ and $t' \twoheadrightarrow \cdot \blacktriangleright v$. Therefore, $(t'_1, \ldots, t'_n) \in \rho(\mathcal{O})$, and $(t_1, \ldots, t_n) \blacktriangleright (t'_1, \ldots, t'_n)$. Hence, $\rho(\delta)$ indeed forms an observation. □

Example 121 Consider the defined function symbol

$$\mathsf{elementAt} : \mathsf{STREAM}_{\mathsf{NAT}} \times \mathsf{NAT} \to \mathsf{NAT}$$

with the rules

$$\begin{aligned} \mathsf{elementAt}(n : x, 0) &\to n \\ \mathsf{elementAt}(n : x, \mathsf{s}\, m) &\to \mathsf{elementAt}(x, m) \end{aligned}$$

Let $C = \lambda s.\, \bot_{\mathsf{NAT}} : s$ and define $\gamma \in \mathcal{O}^{\mathsf{STREAM_{NAT}} \times \mathsf{NAT}}$ by

$$\gamma = \{(C^i(t : \bot_{\mathsf{STREAM_{NAT}}}), \ulcorner i \urcorner) \mid i \in \mathbf{N}, t \in \mathcal{V}^{\star \mathsf{NAT}}\}$$

and let $\rho(\delta) = \gamma$ for every $\delta \in \mathcal{O}^{\mathsf{NAT}}$. Then, ρ forms an observation requirement function, as we have

$$\mathsf{elementAt}(C^i(t : \bot_{\mathsf{STREAM_{NAT}}}), \ulcorner i \urcorner) \to^i \mathsf{elementAt}(t : \bot_{\mathsf{STREAM_{NAT}}}, 0) \to t$$

for every $i \in \mathbf{N}$. Note that this γ cannot be given as a cartesian product. ⊡

3.3.3 Gauge

By looking at the former part of the proof of Theorem 120, one may notice that it is not necessary to have $\rho(\delta)$ for *every* observation δ; if ρ is defined for some observations which are enough to specify each constructor normal form, then it already suffices to ensure productivity of a defined function symbol. Thus, the notion of gauge arises.*

Definition 122 Let S be a sort. Then, a *gauge* of sort S is an infinite sequence $\langle \gamma_k \rangle_{k \in \mathbf{N}}$ of observations of sort S such that $\gamma_0 = \{\bot_S\}$ and

$$\bigcap_{k \in \mathbf{N}} [\![\gamma_k]\!] = \mathcal{V}^{\star S}$$

Given a gauge $\langle \gamma_k \rangle$ of sort S, we can 'measure' terms of sort S by

$$\lfloor t \rfloor_\gamma = \sup\{k \in \mathbf{N} \mid t \in [\![\gamma_k]\!]\}$$

We call this $\lfloor t \rfloor_\gamma$ the *γ-quantity* of t. Note that $\lfloor t \rfloor_\gamma \in \overline{\mathbf{N}}$ (we have $\lfloor t \rfloor_\gamma = \infty$ if the set $\{k \in \mathbf{N} \mid t \in [\![\gamma_k]\!]\}$ is unbounded). Moreover, we define the *potential γ-quantity* of t, written $[\![t]\!]_\gamma$, by $[\![t]\!]_\gamma = \sup\{\lfloor s \rfloor_\gamma \mid t \twoheadrightarrow s\}$. ◪

Proposition 123 *In a quasi-productive algorithmic system, a defined function symbol $f : S_1 \times \cdots \times S_n \to T$ is productive if f admits a 'partial' observation requirement function $\rho : \{\gamma_k \mid k \in \mathbf{N}\} \to \mathcal{O}^{S_1 \times \cdots \times S_n}$, where $\langle \gamma_k \rangle_{k \in \mathbf{N}}$ is a gauge of the sort T.*

Proof: Analogous to the former part of the proof of Theorem 120. Use $\{\gamma_k \mid k \in \mathbf{N}\}$ instead of δ_k. □

Following the quantitative analysis on stream specifications [17], we fix a gauge for each sort, which we shall call 'canonical coinductive gauge'.

*The word 'gauge' has many meanings, most related to a certain measure or measurement, thickness or size. In scientific (physical or mathematical) contexts it has also some connotations with coordinate systems, and measurements of observables.

Definition 124 Define *coinductive truncation* $\mathrm{trunC} : \mathcal{T} \times \mathbf{N} \to \mathcal{P}$ as $\mathrm{trunC}(t, k)$ given by the result of replacing every subterm at position p such that $\|p\|_t = k$ by $\bot$, where $\|p\|_t$, the coinductive depth of p in t, as given in Definition 72. Formally,

$$\begin{aligned} \mathrm{trunC}(t, 0) &\equiv \bot_{\mathrm{srt}(t)} \\ \mathrm{trunC}(f(t_1, \ldots, t_n), j+1) &\equiv f(t'_1, \ldots, t'_n) \qquad (j \in \mathbf{N}) \end{aligned}$$

where

$$t'_i \equiv \mathrm{trunC}(t_i, j + \chi^{\mathbf{I}}_{\mathrm{in}}(f, i)) \qquad (i = 1, \ldots, n)$$

Let S be a sort. Then, a *canonical coinductive gauge* $\langle \kappa_k \rangle^S$, of sort S, is given by $\kappa^S_k = \{\mathrm{trunC}(t, k) \mid t \in \mathcal{V}^{\star S}\}$. ◿

Example 125 In particular, if a sort S is constructed by only inductive sorts, viz. 'purely inductive', then we have $\kappa^S_k = \mathcal{V}^{\star S}$ for every $k > 0$. See Example 115 for example, noticing that NAT is a purely inductive sort. ⊡

Example 126 For the stream of purely inductive objects S given as

$$\mathsf{STREAM}_S :\overset{\mathbf{C}}{=} \mathsf{cons}^S(S, \mathsf{STREAM}_S)$$

we have

$$\kappa^{\mathsf{STREAM}_S}_k = \{t_0 : \ldots : t_{k-1} : \bot_{\mathsf{STREAM}_S} \mid t_0, \ldots, t_{k-1} \in \mathcal{V}^{\star S}\}$$

which coincides with the quantitative analysis studied in [17]. ⊡

Next, we consider constructing observation requirement functions by means of gauges.

Definition 127 Given a gauge γ^S for each sort S and a defined function symbol $f : S_1 \times \cdots \times S_n \to T$, a *$\gamma$-requirement function* for f is a function $\rho : \mathbf{N} \to \mathbf{N}^n$ such that $\llbracket f(t_1, \ldots, t_n) \rrbracket_\gamma \geqslant k$ whenever $\lfloor t_i \rfloor_\gamma \geqslant (\rho(k))_i$ for every applicable i.

A defined symbol f is *γ-productive* if there exists a γ-requirement function for f. ◿

Proposition 128 *Given gauges $\langle \gamma^S \rangle_{S \in \mathbb{S}}$, every γ-productive defined symbol is productive.*

Proof: Immediately from Proposition 123. □

3.3.4 Pre-requirement and κ-requirements

It is much easier to give a γ-requirement function than constructing the whole observation requirement. However, given a function $\mathbf{N} \to \mathbf{N}^n$, it still requires much work to check if a function forms a γ-requirement function, because the definition of γ-requirement function refers to reachability of terms by reduction. For an adequate inductive height estimation or an adequate root activity estimation in preceding sections, we were able to check its adequacy relatively easily by checking some inequalities. We wish to have a criterion for CN, as easy as those for DN and WN. In this subsection, we give a criterion to ensure that a family of functions indeed form κ-requirement functions.

Definition 129 A *pre-requirement* $\boldsymbol{\rho}$ consists of a function $\rho^f : \mathbf{N} \to \mathbf{N}^{\mathrm{ar}(f)}$ for each defined symbol $f \in \mathcal{D}$, that is weakly monotonic with respect to the pointwise ordering on $\mathbf{N}^{\mathrm{ar}(f)}$. ◿

We write $\rho^f_i(k)$ to denote $(\rho^f(k))_i$, i.e. $\rho^f(k) = (\rho^f_1(k), \ldots, \rho^f_{\mathrm{ar}(f)}(k))$.

Definition 130 Given an n-ary defined symbol f, the *redex-requirement* of f, written $\mathrm{rdx}(f)$, is given by $\mathrm{rdx}(f) = (k_1, \ldots, k_n)$, where

$$k_i = \sup\{\|i \cdot p\|_l + 1 \mid (l, r) \in \mathcal{R},\ l(\epsilon) = f,\ l(i \cdot p) \notin \mathcal{X}\}$$

Roughly speaking, the redex-requirement indicates the minimal sufficient κ-quantities of $(t_1, \ldots, t_n)$ so that $f(t_1, \ldots, t_n)$ forms a redex.

A pre-requirement $\boldsymbol{\rho}$ is *redex-consistent* if $\rho^f(1) \geqslant \mathrm{rdx}(f)$, pointwise, for every $f \in \mathcal{D}$. ◿

Definition 131 Let $\boldsymbol{\rho}$ be a pre-requirement. Given a term t and $k \in \mathbf{N}$, *requirement labeling* $\mathrm{lbl}_{\boldsymbol{\rho}}(t; k) : \mathrm{dom}(t) \to \mathbf{N}$ is defined by

$$\mathrm{lbl}_{\boldsymbol{\rho}}(t; k)(\epsilon) = k$$
$$\mathrm{lbl}_{\boldsymbol{\rho}}(t; k)(p \cdot n) = \zeta(\mathrm{lbl}_{\boldsymbol{\rho}}(t; k)(p), t(p), n)$$

where $\zeta : \mathbf{N} \times \Sigma \times \mathbf{N} \to \mathbf{N}$ is defined by

$$\zeta(h, f, i) = \begin{cases} 0 & (\text{if } h = 0) \\ h - \chi^{\mathbf{C}}_{\mathrm{in}}(f, i) & (\text{if } h > 0 \text{ and } f \in \mathcal{C}) \\ \rho^f_i(h)) & (\text{if } h > 0 \text{ and } f \in \mathcal{D}) \end{cases}$$

◿

To state the intuition of the above definition, requirement labeling $\mathrm{lbl}_{\boldsymbol{\rho}}(t; k)$ indicates the required κ-quantity of each subterm of t to compute t up to the coinductive depth k, provided that the pre-requirement indeed induces κ-requirement functions.

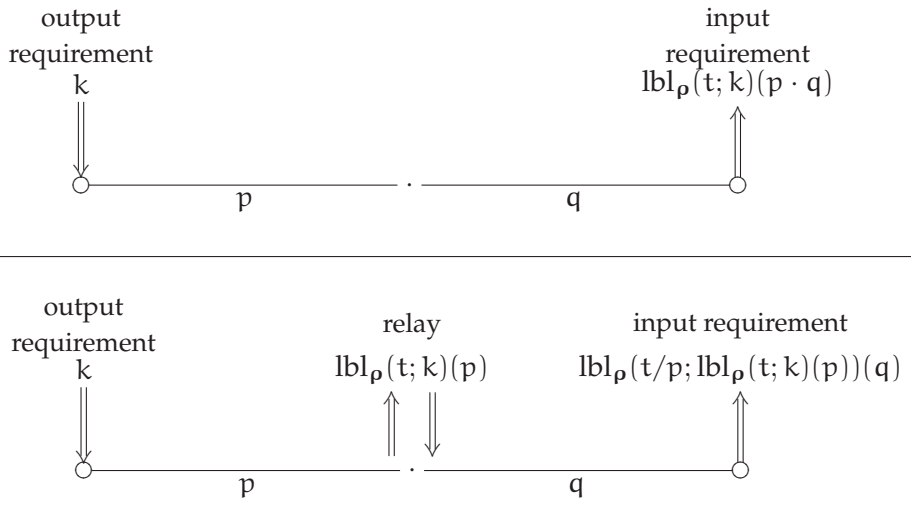


Figure 21: *Proposition 132*

Proposition 132 *Let* $\boldsymbol{\rho}$ *be a pre-requirement.*

$$\mathrm{lbl}_{\boldsymbol{\rho}}(t;k)(p\cdot q)=\mathrm{lbl}_{\boldsymbol{\rho}}(t/p;\mathrm{lbl}_{\boldsymbol{\rho}}(t;k)(p))(q)$$

for every $t\in\mathcal{T}$, $k\in\mathbf{N}$, *and* $p,q\in\mathbf{N}_+^*$ *such that* $p\cdot q\in\mathrm{dom}(t)$.

Proof: Follows from the above definition. See also Figure 21. □

Definition 133 A pre-requirement $\boldsymbol{\rho}$ is *reduction-consistent* if

$$\mathrm{lbl}_{\boldsymbol{\rho}}(l;k)(p)\geqslant\mathrm{lbl}_{\boldsymbol{\rho}}(r;k)(q)$$

for every $(l,r)\in\mathcal{R}$, $p\in\mathrm{dom}(l)$, $q\in\mathrm{dom}(r)$ such that $l(p)=r(q)\in\mathcal{X}$, and for every $k\in\mathbf{N}$. ◪

Definition 134 A pre-requirement is *red*^{2}*-consistent* if it is *red*ex-consistent and *red*uction-consistent. ◪

Example 135 Again consider the systems

$$f(n:m:x)\to n:m:f(f(x)) \tag{A}$$

and

$$f(n:m:x)\to n:f(f(x)) \tag{B}$$

as in Subsection 3.3.2. Let $\rho^f(k)=\mathrm{floor}((k+1)/2)\times 2$ in the system (A).[†] It satisfies redex-consistence

$$\rho^f(1)\geqslant 2$$

[†]The function floor is defined by $\mathrm{floor}(x)=\max\{n\in\mathbf{N}\mid n\leqslant x\}$.

and reduction-consistence

$$\begin{aligned}\rho^{\mathsf{f}}(k) &\geqslant k\\ \rho^{\mathsf{f}}(k)-2 &\geqslant \rho^{\mathsf{f}}(\rho^{\mathsf{f}}(k-2))\end{aligned}$$

Thus, this pre-requirement is red^2-consistent.

On the other hand, a red^2-consistent pre-requirement ρ^{f} for the system (B) has to satisfy

$$\begin{aligned}\rho^{\mathsf{f}}(1) &\geqslant 2\\ \rho^{\mathsf{f}}(k) &\geqslant k\\ \rho^{\mathsf{f}}(k)-2 &\geqslant \rho^{\mathsf{f}}(\rho^{\mathsf{f}}(k-1))\end{aligned}$$

which leads to contradiction:

$$\begin{aligned}\rho^{\mathsf{f}}(2)-2 &\geqslant \rho^{\mathsf{f}}(\rho^{\mathsf{f}}(2-1))\\ &= \rho^{\mathsf{f}}(\rho^{\mathsf{f}}(1))\\ &\geqslant \rho^{\mathsf{f}}(2)\end{aligned}$$

Thus, the system (B) admits no red^2-consistent pre-requirement. ⊡

Example 136 We show a red^2-consistent pre-requirement for the system HAM. Let $\boldsymbol{\rho}$ be a pre-requirement. If $\boldsymbol{\rho}$ is redex-consistent, $\boldsymbol{\rho}$ must satisfy the following inequalities

$$\begin{aligned}\rho^{+}(1) &\geqslant (0,1)\\ \rho^{\times}(1) &\geqslant (0,1)\\ \rho^{\mathsf{sca}}(1) &\geqslant (1,0)\\ \rho^{\mathsf{cmp}}(1) &\geqslant (1,1)\\ \rho^{\mathsf{mrg}}(1) &\geqslant (1,1)\\ \rho^{\mathsf{aux}}(1) &\geqslant (1,1,1)\end{aligned}$$

pointwise. And, the inequalities shown in Table 22 are required for reduction-consistence of $\boldsymbol{\rho}$, where

$$h^{-} = \begin{cases} h-1 & (\text{if } h > 0)\\ 0 & (\text{if } h = 0)\end{cases}$$

One may get scared off by 33 inequations in total, however, at least for this system, it is not so difficult to reach a pre-requirement

$$\begin{aligned}\rho^{+}(k) &= (k, k\wedge 1)\\ \rho^{\times}(k) &= (k, k\wedge 1)\\ \rho^{\mathsf{sca}}(k) &= (k, k\wedge 1)\\ \rho^{\mathsf{mrg}}(k) &= (k, k)\\ \rho^{\mathsf{aux}}(k) &= (k\wedge 1, k, k)\end{aligned}$$

Observe that ρ is red^2-consistent. ⊡

However, a red^2-consistent pre-requirement does not always form κ-requirements. For the easiest counterexample, consider the one-rule system

$$\mathsf{f} \to \mathsf{f}$$

There is also a counterexample which is quasi-productive;

$$\begin{aligned} \mathsf{id}(n : x) &\to n : \mathsf{id}(x) \\ \mathsf{f} &\to \mathsf{id}(\mathsf{f}) \end{aligned}$$

3.3.5 Dependency

In the previous subsection, we have seen that red^2-consistence of the pre-requirement is still not enough to induce κ-requirement functions. In order to fix this problem, we introduce the notion of requirement dependency.

Definition 137 Let ρ be a pre-requirement. Then, a *requirement dependency relation* is a relation on $\mathcal{D} \times \mathbf{N}$ given as $(f, k) \rightarrowtriangle (g, h)$ iff $k > 0$ and there exist $(l, r) \in \mathcal{R}$ and $p \in \mathrm{dom}(r)$ such that $l(\epsilon) = f$, $r(p) = g$, and $\mathrm{lbl}_\rho(r; k)(p) = h$. Moreover, in this case, we write $(f, k) \rightarrowtriangle^N (g, h)$ if there exists $q \lhd p$ such that $r(q) \in \mathcal{D}$, viz. $g = r(p)$ is a nested occurrence of a defined function symbol; $(f, k) \rightarrowtriangle^O (g, h)$ otherwise, viz. g is an outermost occurrence of a defined function symbol. ◪

Example 138 Roughly speaking, a pair $(f, k) \in \mathcal{D} \times \mathbf{N}$ represents the task of normalizing any term of the form $f(\dots)$ up to the depth k. For a more concrete intuition, consider the defined symbols Ham and Ham2, which mutually depend:

$$\begin{aligned} \mathsf{Ham} &\to \mathsf{s}\,0 : \mathsf{mrg}(\mathsf{mrg}(\mathsf{Ham2}, \mathsf{Ham3}), \mathsf{Ham5}) \\ \mathsf{Ham2} &\to \mathsf{sca}(\mathsf{Ham}, \mathsf{s}\,\mathsf{s}\,0) \end{aligned}$$

With the pre-requirement ρ as shown in Example 136, requirement labeling $\mathrm{lbl}_\rho(-; k)$ on the righthand-side of the above rules is as indicated in the superscripts:

$$\begin{aligned} &\mathsf{cons}^{\mathsf{NAT}^k}(\mathsf{s}^k0^k, \mathsf{mrg}^{k-1}(\mathsf{mrg}^{k-1}(\mathsf{Ham2}^{k-1}, \mathsf{Ham3}^{k-1}), \mathsf{Ham5}^{k-1})) \\ &\mathsf{sca}^k(\mathsf{Ham}^k, \mathsf{s}^1\mathsf{s}^1 0^1). \end{aligned}$$

provided $k > 0$. Thus, we have

$$(\mathsf{Ham2}, k+1) \rightarrowtriangle^N (\mathsf{Ham}, k+1) \rightarrowtriangle^N (\mathsf{Ham2}, k)$$

for every $k \in \mathbf{N}$. ⊡

rule	variable	inequalities
$n + 0 \to n$	n	$\rho_1^+(k) \geqslant k$
$n + s\,m \to s(n + m)$	n	$\rho_1^+(k) \geqslant \rho_1^+(k)$
	m	$\rho_2^+(k) \geqslant \rho_2^+(k)$
$n \times s\,m \to (n \times m) + n$	n	$\rho_1^\times(k) \geqslant \rho_1^\times(\rho_1^+(k))$
		$\rho_1^\times(k) \geqslant \rho_2^+(k)$
	m	$\rho_2^\times(k) \geqslant \rho_2^\times(\rho_1^+(k))$
$\mathsf{sca}(n : x, m) \to (n \times m) : \mathsf{sca}(x, m)$	n	$\rho_1^{\mathsf{sca}}(k) \geqslant \rho_1^\times(k)$
	m	$\rho_2^{\mathsf{sca}}(k) \geqslant \rho_2^\times(k)$
		$\rho_2^{\mathsf{sca}}(k) \geqslant \rho_2^{\mathsf{sca}}(k^-)$
	x	$\rho_1^{\mathsf{sca}}(k)^- \geqslant \rho_1^{\mathsf{sca}}(k^-)$
$\mathsf{cmp}(s\,n, s\,m) \to \mathsf{cmp}(n, m)$	n	$\rho_1^{\mathsf{cmp}}(k) \geqslant \rho_1^{\mathsf{cmp}}(k)$
	m	$\rho_2^{\mathsf{cmp}}(k) \geqslant \rho_2^{\mathsf{cmp}}(k)$
$\mathsf{mrg}(n : x, m : y)$	n	$\rho_1^{\mathsf{mrg}}(k) \geqslant \rho_1^{\mathsf{cmp}}(\rho_1^{\mathsf{aux}}(k))$
$\to \mathsf{aux}(\mathsf{cmp}(n, m), n : x, m : y)$		$\rho_1^{\mathsf{mrg}}(k) \geqslant \rho_2^{\mathsf{aux}}(k)$
	m	$\rho_2^{\mathsf{mrg}}(k) \geqslant \rho^{\mathsf{cmp}}(\rho_1^{\mathsf{aux}}(k))$
		$\rho_2^{\mathsf{mrg}}(k) \geqslant \rho_3^{\mathsf{aux}}(k)$
	x	$\rho_1^{\mathsf{mrg}}(k)^- \geqslant \rho_2^{\mathsf{aux}}(k)^-$
	y	$\rho_2^{\mathsf{mrg}}(k)^- \geqslant \rho_3^{\mathsf{aux}}(k)^-$
$\mathsf{aux}(\mathsf{eq}, n : x, m : y) \to n : \mathsf{mrg}(x, y)$	n	$\rho_2^{\mathsf{aux}}(k) \geqslant k$
	x	$\rho_2^{\mathsf{aux}}(k)^- \geqslant \rho_1^{\mathsf{mrg}}(k)^-$
	y	$\rho_3^{\mathsf{aux}}(k)^- \geqslant \rho_2^{\mathsf{mrg}}(k)^-$
$\mathsf{aux}(\mathsf{gt}, x, m : y) \to m : \mathsf{mrg}(x, y)$	m	$\rho_3^{\mathsf{aux}}(k) \geqslant k$
	x	$\rho_2^{\mathsf{aux}}(k)^- \geqslant \rho_1^{\mathsf{mrg}}(k)^-$
	y	$\rho_3^{\mathsf{aux}}(k)^- \geqslant \rho_2^{\mathsf{mrg}}(k)^-$
$\mathsf{aux}(\mathsf{lt}, n : x, y) \to n : \mathsf{mrg}(x, y)$	n	$\rho_2^{\mathsf{aux}}(k) \geqslant k$
	x	$\rho_2^{\mathsf{aux}}(k)^- \geqslant \rho_1^{\mathsf{mrg}}(k)^-$
	y	$\rho_3^{\mathsf{aux}}(k)^- \geqslant \rho_2^{\mathsf{mrg}}(k)^-$

Table 22: *Inequalities for reduction-consistence*

Theorem 139 *A pre-requirement induces κ-requirement function for each defined function symbol if the following conditions are satisfied.*

1. *The system is quasi-productive.*
2. *The pre-requirement is red^2-consistent.*
3. *There exists a $\twoheadrightarrow^{\mathbf{N}} \mid \twoheadrightarrow^{\mathbf{O}}$-compatible well-founded quasiorder on $\mathcal{D} \times \mathbf{N}$.*

Proof: Suppose that the system is quasi-productive. By Theorems 90 and 97, we can find respectively a $\downarrow^{\mathbf{I}} \mid \downarrow^{\mathbf{C}} \rightarrow$-compatible well-founded quasiorder $\preccurlyeq_{DN}$ and a $\rightarrow_\epsilon \mid \rightarrow$-compatible well-founded quasiorder $\preccurlyeq_{WN}$ on $\mathcal{G}^\star$. Let $\boldsymbol{\rho}$ be a red^2-consistent pre-requirement, and $\preccurlyeq_{CN}$ be a $\twoheadrightarrow^{\mathbf{N}} \mid \twoheadrightarrow^{\mathbf{O}}$-compatible well-founded quasiorder on $\mathcal{D} \times \mathbf{N}$ where $\twoheadrightarrow^{\mathbf{N}}$ and $\twoheadrightarrow^{\mathbf{O}}$ are derived from $\boldsymbol{\rho}$.

Now, let $\Xi = \{(t, k) \in \mathcal{G}^\star \times \mathbf{N} \mid t(\epsilon) \in \mathcal{D}\}$. Roughly speaking, $(t, k) \in \Xi$ represents the task of normalizing the term t up to the coinductive depth n. So, we say that (t, k) is *irregular* if $\llbracket t \rrbracket_\kappa < k$ despite $\lfloor t/i \rfloor_\kappa \geqslant \rho_i^{t(\epsilon)}(k)$ for every applicable i; *regular* otherwise. From the definition of requirement functions, to show that every $(t, n) \in \Xi$ is regular will suffice. Let $\preccurlyeq_\Xi$ be a quasiorder on Ξ defined as

$$\begin{array}{l}
(t,k) \preccurlyeq_\Xi (s,h) \text{ iff} \\
(t(\epsilon),k) \prec_{CN} (s(\epsilon),h) \text{ or} \\
[(t(\epsilon),k) \preccurlyeq_{CN} (s(\epsilon),h) \text{ and } [t \prec_{DN} s \text{ or} \\
\qquad\qquad [t \preccurlyeq_{DN} s \text{ and } [k < h \text{ or} \\
\qquad\qquad\qquad [k \leqslant h \text{ and } t \preccurlyeq_{WN} s]]]]]
\end{array}$$

Intuitively, this order is a conjunction of $\preccurlyeq_{CN}$, $\preccurlyeq_{DN}$, $\preccurlyeq_{WN}$, and $<_{\mathbf{N}}$ with the lexicographic priority. Observe that $\langle \Xi, \preccurlyeq_\Xi \rangle$ is well-founded.

In order to prove that every $(t, k) \in \Xi$ is regular, using induction, we fix $(t, k) \in \Xi$ and assume that (s, h) is regular whenever $(s, h) \prec_\Xi (t, k)$. We will show that (t, k) is regular. Suppose $\lfloor t/i \rfloor_\kappa \geqslant \rho_i^{t(\epsilon)}(k)$ and $k > 0$; otherwise regularity of (t, k) is trivial. Since the pre-requirement $\boldsymbol{\rho}$ is redex-consistent, with $k > 0$, we can find $(l, r) \in \mathcal{R}$ and $\sigma : \mathcal{X} \to \mathcal{G}^\star$ such that

$$l\sigma \equiv t \rightarrow_\epsilon s \equiv r\sigma$$

Since $\llbracket t \rrbracket_\kappa = \llbracket s \rrbracket_\kappa$, to show $\llbracket s \rrbracket_\kappa \geqslant k$ will suffice. In order to show this by contradiction, assume $\llbracket s \rrbracket_\kappa < k$. Now we construct an infinite sequence $\langle p_i \rangle_{i \in \mathbf{N}}$ of positions in r, satisfying

1. $|p_i| = i$, and
2. $\llbracket s/p_i \rrbracket_\kappa < \mathrm{lbl}_\kappa(s; k)(p_i)$

for every i. Clearly, the first condition contradicts the fact that r is finite. Let $p_0 = \epsilon$. We have $[\![s/\epsilon]\!]_\kappa = [\![s]\!]_\kappa < k = \mathrm{lbl}_\kappa(s;k)(\epsilon)$. Now, suppose p_i is already defined to satisfy the above conditions. In order to define p_{i+1}, we perform the following case analysis on $r(p_i)$.

- **The case** $r(p_i) \in \mathcal{D}$. We will show $(s/p_i, \mathrm{lbl}_\kappa(s;k)(p_i)) \prec_\Xi (t,k)$. We have $(s(p_i), \mathrm{lbl}_\kappa(s;k)(p_i)) \preccurlyeq_{CN} (t(\epsilon),k)$ by $\twoheadrightarrow^N \mid \twoheadrightarrow^O$-compatibility. Since $\preccurlyeq_{DN}$ is $\searrow^{\mathbf{I}} \mid \searrow^{\mathbf{C}} \to$-compatible, we have $s/p_i \preccurlyeq_{DN} s \preccurlyeq_{DN} t$.
  - If $r(p_i)$ is an outermost occurrence of a defined symbol, i.e. there exists no $q \lhd p_i$ such that $r(q) \in \mathcal{D}$, then we have
    $$s(\searrow^{\mathbf{I}} \cup \searrow^{\mathbf{C}})^* s/(p_i) \qquad (*)$$
    If there exists some $\searrow^{\mathbf{I}}$ steps in $(*)$, then we have $s(p_i) \prec_{DN} s$ by $\searrow^{\mathbf{I}} \mid \searrow^{\mathbf{C}}$-compatibility, implying $(s/p_i, \mathrm{lbl}_\kappa(s;k)(p_i)) \prec_\Xi (t,k)$.
    If there exists no $\searrow^{\mathbf{I}}$ step but some $\searrow^{\mathbf{C}}$ steps in $(*)$, then we have $\mathrm{lbl}_\kappa(s;k)(p_i) < k$, implying $(s/p_i, \mathrm{lbl}_\kappa(s;k)(p_i)) \prec_\Xi (t,k)$.
    If there exists no $\searrow^{\mathbf{I}}$ step nor $\searrow^{\mathbf{C}}$ step, i.e. $p = \epsilon$, then we have $\mathrm{lbl}_\kappa(s;l)(p_i) = k$, and $s/p_i \prec_{WN} t$ from $\to_\epsilon \mid \varnothing$-compatibility of $\preccurlyeq_{WN}$.
    Therefore, $(s/p_i, \mathrm{lbl}_\kappa(s;k)(p_i)) \prec_\Xi (t,k)$.
  - If $r(p_i)$ is a nested occurrence of a defined symbol, then we have $(s(p_i), \mathrm{lbl}_\kappa(s;k)(p_i)) \prec_{CN} (t(\epsilon),k)$ from $\twoheadrightarrow^N \mid \varnothing$-compatibility of $\preccurlyeq_{CN}$, implying $(s/p_i, \mathrm{lbl}_\kappa(s;k)(p_i)) \prec_\Xi (t,k)$.

  In each case we have $(s/p_i, \mathrm{lbl}_\kappa(s;k)(p_i)) \prec_\Xi (t,k)$, and thus the $(s/p_i, \mathrm{lbl}_\kappa(s;k)(p_i))$ is regular by the induction hypothesis. Since we have $[\![s/p_i]\!]_\kappa < \mathrm{lbl}_\kappa(s;k)(p_i)$, by the definition of regularity, we can find some j such that $p_i \cdot j \in \mathrm{dom}(r)$ and
  $$\begin{aligned}[\![s/p_i \cdot j]\!]_\kappa &< \rho_j^{s(p_i)}(\mathrm{lbl}_\kappa(s;k)(p_i)) \\ &= \mathrm{lbl}_\kappa(s;k)(p_i \cdot j)\end{aligned}$$
  Let $p_{i+1} = p_i \cdot j$.

- **The case** $r(p_i) \in \mathcal{C}$. We have
  $$[\![s/p_i]\!]_\kappa = \inf\{[\![s/p_i \cdot j]\!]_\kappa + \chi_{\mathrm{in}}^{\mathbf{C}}(s(p_i), j) \mid j = 1, \ldots, \mathrm{ar}(s(p))\}$$
  By the assumption $[\![s/p_i]\!]_\kappa < \mathrm{lbl}_\kappa(s;k)(p_i)$, we can find some j such that
  $$\begin{aligned}[\![s/p_i \cdot j]\!]_\kappa &< \mathrm{lbl}_\kappa(s;k)(p_i) - \chi_{\mathrm{in}}^{\mathbf{C}}(s(p_i), j) \\ &= \mathrm{lbl}_\kappa(s;k)(p_i \cdot j)\end{aligned}$$
  Let $p_{i+1} = p_i \cdot j$.

- **The case** $r(p_i) \in \mathcal{X}$. We can find some $h \in \mathbf{N}$ and $q \in \mathbf{N}_+^*$ such that $h \cdot q \in \mathrm{dom}(l)$ and $l(h \cdot q) = r(p_i)$. Then, we have

$$
\begin{array}{ll}
& [\![s/p_i]\!]_\kappa \\
= & [\![t/h \cdot q]\!]_\kappa & (\text{since } s/p_i \equiv t/h \cdot q \equiv \sigma(r(p_i))) \\
\geqslant & \mathrm{lbl}_\kappa(t/h; [\![t/h]\!]_\kappa)(q) & (\text{by definitions of lbl and potential } \kappa\text{-quantity}) \\
\geqslant & \mathrm{lbl}_\kappa(t/h; \rho_h^{t(\epsilon)}(k))(q) & (\text{by the assumption } t/h \leqslant \rho_h^{t(\epsilon)}(k) \text{ and monotonicity of } \boldsymbol{\rho}) \\
= & \mathrm{lbl}_\kappa(t; k)(h \cdot q) & (\text{by definition of lbl}) \\
= & \mathrm{lbl}_\kappa(l; k)(h \cdot q) & (\text{by coincidence between } l \text{ and } t \equiv l\sigma \text{ up to the position } h \cdot q) \\
= & \mathrm{lbl}_\kappa(r; k)(p_i) & (\text{by reduction-consistency of the pre-requirement } \boldsymbol{\rho}) \\
= & \mathrm{lbl}_\kappa(s; k)(p_i) & (\text{by coincidence between } r \text{ and } s \equiv r\sigma \text{ up to the position } p_i)
\end{array}
$$

  which contradicts the assumption $[\![s/p_i]\!]_\kappa < \mathrm{lbl}_\kappa(s; k)(p_i)$. Hence, this case cannot occur. □

Thus, we have given a criterion for CN. But, a $\twoheadrightarrow^N \mid \twoheadrightarrow^O$-compatible well-founded quasiorder on $\mathcal{D} \times \mathbf{N}$ is a rather strong requirement; this criterion does not help to prove CN for the system HAM. We have

$$(\times, k) \twoheadrightarrow^N (\times, k)$$

from the rule $n \times s\, m \to (n \times m) + n$. Therefore, there could exist no $\twoheadrightarrow^N \mid \varnothing$-compatible well-founded quasiorder on $(\mathcal{D}, \mathbf{N})$. Curiously, the module consisting of $+$ and $\times$, which is even strongly CN, causes the problem. In fact, productivity of this module

$$
\begin{aligned}
n + 0 &\to n \\
n + s\, m &\to s(n + m) \\
n \times 0 &\to 0 \\
n \times s\, m &\to (n \times m) + n
\end{aligned}
$$

relies on the inductive construction of NAT, as discussed in the Introduction. Thus, we ramify the *tasks* $(\mathcal{D}, \mathbf{N})$ indexed by estimated inductive heights of arguments.

Definition 140 A *task* is a tuple $(f, k, \alpha_1, \ldots, \alpha_{\mathrm{ar}(f)})$, where $f \in \mathcal{D}$, $k \in \mathbf{N}$, and $\alpha_1, \ldots, \alpha_{\mathrm{ar}(f)} \in \boldsymbol{\Omega}$. Given an adequate inductive height estimator $\mathbf{H}$ and a pre-requirement $\boldsymbol{\rho}$, a *ramified requirement dependency relation* is defined as

$$(f, k, \alpha_1, \ldots, \alpha_{\mathrm{ar}(f)}) \twoheadrightarrow (g, h, \beta_1, \ldots, \beta_{\mathrm{ar}(g)})$$

iff there exists $(l, r) \in \mathcal{R}$, $p \in \mathrm{dom}(r)$, and $\zeta : \mathcal{X} \to \mathbf{\Omega}$ such that $l(\epsilon) = f$, $r(p) = g$, $\mathrm{lbl}_{\boldsymbol{\rho}}(r; k)(p) = h$, $[\![l/i]\!]^{\zeta}_{\mathbf{H}} = \alpha_i$, and $[\![r/p \cdot j]\!]^{\zeta}_{\mathbf{H}} = \beta_j$ for every applicable i and j. We divide $\rightarrowtriangle$ into $\rightarrowtriangle^{N}$ and $\rightarrowtriangle^{O}$ analogously as Definition 137. ◿

Then, the refined version of Theorem 139 follows:

Theorem 141 *Let* $\boldsymbol{\rho}$ *be a pre-requirement, and let* $\mathbf{H}$ *be an adequate inductive height estimator. Then,* $\boldsymbol{\rho}$ *forms* κ*-requirement functions for each defined function symbol if the following conditions are satisfied.*

1. *The system is* WN.
2. *The pre-requirement is red*^{2}*-consistent.*
3. *There exists a* $\rightarrowtriangle^{N} \mid \rightarrowtriangle^{O}$*-compatible well-founded quasiorder on the tasks.*

Proof: Analogously as Theorem 139. □

Corollary 142 *Let* $\boldsymbol{\rho}$ *be a pre-requirement, and let* $\mathbf{H}$ *be an adequate inductive height estimator. If the conditions as in the above theorem are satisfied, then the system is productive.*

Proof: The existence of the adequate inductive height estimator $\mathbf{H}$ ensures DN of the system. Since WN of the system is already supposed, the system is quasi-productive. From the above theorem, every defined symbol is κ-productive, and therefore productive by Proposition 128. Thus, by Lemma 109, the system is productive. □

Example 143 Let $\mathbf{H}$ and $\boldsymbol{\rho}$ be an inductive height estimator as shown in Example 101 and a pre-requirement as in Example 136, respectively. Then, the derived ramified requirement dependency is as follows.

$$
\begin{aligned}
(+, k, \alpha, \beta + 1) &\rightarrowtriangle^{O} (+, k, \alpha, \beta)\\
(\times, k, \alpha, \beta + 1) &\rightarrowtriangle^{O} (+, k, \alpha \cdot \beta, \alpha)\\
(\times, k, \alpha, \beta + 1) &\rightarrowtriangle^{N} (\times, k, \alpha, \beta)\\
(\mathsf{sca}, k, \alpha + 1, \beta) &\rightarrowtriangle^{O} (\times, k, \alpha, \beta)\\
(\mathsf{sca}, k, \alpha, \beta) &\rightarrowtriangle^{O} (\mathsf{sca}, k - 1, \alpha, \beta)\\
(\mathsf{cmp}, k, \alpha + 1, \beta + 1) &\rightarrowtriangle^{O} (\mathsf{cmp}, k \wedge 1, \alpha, \beta)\\
(\mathsf{mrg}, k, \alpha, \beta) &\rightarrowtriangle^{N} (\mathsf{aux}, k, 0, \alpha, \beta)\\
(\mathsf{aux}, k, \alpha, \beta) &\rightarrowtriangle^{O} (\mathsf{mrg}, k - 1, \alpha, \beta)\\
(\mathsf{Ham}, k) &\rightarrowtriangle^{O} (\mathsf{mrg}, k - 1, \omega, \omega)\\
(\mathsf{Ham}, k) &\rightarrowtriangle^{N} (\mathsf{Ham2}, k - 1), (\mathsf{Ham3}, k - 1), (\mathsf{Ham5}, k - 1)
\end{aligned}
$$

$$(\mathsf{Ham2}, k) \rightarrow^{O} (\mathsf{sca}, k, \omega, 2)$$
$$(\mathsf{Ham2}, k) \rightarrow^{N} (\mathsf{Ham}, k)$$
$$(\mathsf{Ham3}, k) \rightarrow^{O} (\mathsf{sca}, k, \omega, 3)$$
$$(\mathsf{Ham3}, k) \rightarrow^{N} (\mathsf{Ham}, k)$$
$$(\mathsf{Ham5}, k) \rightarrow^{O} (\mathsf{sca}, k, \omega, 5)$$
$$(\mathsf{Ham5}, k) \rightarrow^{N} (\mathsf{Ham}, k)$$

where $k > 0$, $\alpha, \beta \in \mathbf{\Omega}$.

Now, let

$$+ \;<\; \times \;<\; \mathsf{sca}$$
$$\mathsf{cmp} \;<\; \mathsf{aux} \;<\; \mathsf{mrg} \;<\; \mathsf{Ham} \;<\; \mathsf{Ham2}, \mathsf{Ham3}, \mathsf{Ham5}$$

and introduce the lexicographic order on the task with the priority k, f, $\alpha_1, \ldots, \alpha_n$. Then, that order forms a $\rightarrow^{N} | \rightarrow^{O}$-compatible well-founded quasiorder. Thus, by the above corollary, the system HAM is shown to be productive. ⊡

Remark 144 The order on the task as introduced in the above example is not only $\rightarrow^{N} | \rightarrow^{O}$-compatible but also $\rightarrow | \varnothing$-compatible. That order thus guarantees that the system is CN and WN. ⧈

Remark 145 Note that the system HAM, as far as mixing inductive and coinductive components is concerned, has a very simple structure: streams of natural numbers. So, it is not surprising that other methods exist that can deal in a simpler way with the productivity proof of that system. For example, productivity of the system dealing with Hamming numbers has also been proved by Endrullis et al. [16], using their productivity tool. Another approach might be the method of Zantema [55], using an 'observational term rewriting system'. The problem of whether the n-th entry of the Hamming number sequence is correctly defined is there reduced to the termination problem of the associated observational TRS. That termination problem can then be subjected to the current termination technology for an automatic proof. ⧈

3.4 Conclusion

Table 23 concludes our characterizations of each of the properties DN, WN, and SCN of an algorithmic system. For all the strongly productive algorithmic systems, their strong productivity can be proved with these characterizations.

Table 24 shows criteria for each of the properties DN, WN, and CN of an algorithmic system. For many productive algorithmic systems, their

	characterizing well-founded quasiorder on $\mathcal{G}^\star$	reference
DN	$\searrow^{\mathbf{I}} \mid \searrow^{\mathbf{C}} \rightarrow$-compatible	Theorems 89 and 90
WN	$\rightarrow_\epsilon \mid \rightarrow$-compatible	Theorems 96 and 97
SCN	$\searrow^{\mathcal{D}} \mid \searrow \rightarrow$-compatible	Theorem 106

Table 23: *Survey of characterizations*

	criterion	reference
DN	adequate inductive height estimator	Theorem 92
WN	adequate root activity estimator	Theorem 100
CN	red^2-consistent pre-requirement with $\twoheadrightarrow^{\mathsf{N}} \mid \twoheadrightarrow^{\mathsf{O}}$-compatible well-founded ramified requirement dependency relation	Corollary 142

Table 24: *Survey of criteria*

productivity can be proved with these criteria. It should be remarked that our criterion for CN requires quasi-productivity of the system.

The criterion for CN is so complicated that it remains laborious to ensure productivity. As to future work, we wish to automate the process of finding adequate estimators for DN and CN, and applicable pre-requirements.

The machinery in this thesis was developed not only for streams, but for the most general mixture of inductive and coinductive data and codata. And reaching full generality has led to the present theoretical building, where the most effort had to be spent to the CN property. We point out that our machinery is very modular, and has separated the various concerns, to wit, DN, WN, and CN. So even if one deems the present state of the method for CN to be too complicated, there are still the more manageable method to deduce DN and WN. For these properties, the sufficient criterion that we have presented is indeed just as simple as known recursive path order methods, or lexicographic iterative path order methods. For these properties DN and WN, automatization is in our opinion very feasible. For particular examples, an automated proof of DN and WN can then be completed by an ad hoc proof by some easier method to obtain CN. But again we stress that our presently developed method for CN is fully general, and therefore may serve as a framework or guideline, as a theoretical background, for the future development of simplified methods for particular subclasses of mixed inductive-coinductive specifications, and the corresponding automated tools for such methods.

For our other paradigm running example, the tree ordinals up to var-

ious heights, no ad hoc productivity proof methods are presently known. Here we have streams of streams of streams..., in a well-founded hierarchy. Our closing chapter is devoted to this highly nontrivial mixture of inductive and coinductive (co-)data. Here our fully general machinery is put to the test. Of course, also for this particular ATRS (in its various extensions) one may eventually develop a simpler ad hoc proof. But that does not detract from what we feel is valuable, namely the full generality of the present framework.

Chapter 4

Tree ordinals

In this chapter, we study algorithmic systems for tree ordinals as a paradigmatic example of a construction using interdependent inductive and coinductive sorts. Thereby we increase our insight in the strength and expressivity of the term rewriting framework, in particular, term rewriting notations for large ordinals.

Tree ordinals constitute an economic way to work with ordinals and their arithmetic. The economy is that we do not work with the actual ordinals, but with representations of them that are a sort of thinned out versions. These are known in proof theory as fundamental sequences, which are ω-long sequences (or streams) having as elements natural numbers (finite ordinals) or again fundamental sequences (standing for infinite ordinals). The nesting of such fundamental sequences is only finitely deep, leading to a countably branching, well-founded tree.

To represent a countable branching is not a priori possible in first order term rewriting, at least not in the usual version; but we can just as well work with infinite sequences which can be obtained by iterated pairing, thus staying in the finitely branching framework of first order (possibly infinitely deep) terms.

We will present algorithmic term rewriting systems for three 'landmark ordinals', to wit ϵ_0, the Feferman–Schütte ordinal Γ_0, and the small Veblen ordinal $\theta_{\mathbf{\Omega}^\omega}(0)$, together with the proofs of the productivity of these systems. The functions introduced to construct those large ordinals are all studied in [41], and hence the algorithmic systems presented in this chapter give equivalent constructions as performed ibid. below the representation limit of each system. From the viewpoint of construction of ordinals, this chapter can be related also to [40].

4.1 Representation limit

In this chapter we are concerned with representing large ordinals by finite expressions. When a productive system employing tree ordinals (ORD) is given, we call the ordinal given by

$$\min(\mathbf{\Omega} \setminus \{[\![t]\!] \mid t \in \mathcal{T}^{\mathsf{ORD}}\})$$

viz. the first ordinal which cannot be finitely represented in the given system, the *representation limit* of the sort ORD. Since the set of finite terms is countable, any system has the representation limit less than or equal to ω_1^{CK}, the Church-Kleene ordinal.

Proposition 146 *The representation limit of the system* ORD-AME*, as shown in Table 15, is* ϵ_0.

Proof: Let $A = \{[\![t]\!] \mid t \in \mathcal{T}^{\mathsf{ORD}}\}$ and B be defined by

$$\begin{aligned} B_0 &= \{0, 1, \omega\} \\ B_{n+1} &= B_n \cup \{\alpha + \beta, \alpha \cdot \beta, \alpha^\gamma \mid \alpha, \beta \in B_n,\ \gamma \in B_n \setminus \{0\}\} \end{aligned}$$

and

$$B = \bigcup_{n \in \mathbf{N}} B_n$$

That is, B consists of the ordinals that can be obtained by means of 0, 1, ω, addition, multiplication, and exponentiation. Then we have clearly $A = B$. Since $A = \epsilon_0$ will suffice for the claim, to show $B = \epsilon_0$ will also suffice.

($B \subseteq \epsilon_0$) Clearly $0, 1, \omega < \epsilon_0$. And, if $\alpha, \beta < \epsilon_0$, then $\alpha + \beta$, $\alpha \cdot \beta$, and α^β (when $\beta > 0$) are all smaller than ϵ_0. Thus, by mathematical induction on n, we have $B_n \subseteq \epsilon_0$. Hence, $B \subseteq \epsilon_0$.

($B \supseteq \epsilon_0$) For a proof by contradiction, assume that $B \not\supseteq \epsilon_0$ and let $\alpha = \min(\epsilon_0 \setminus B)$. Then, α must be limit ordinal, because, if $\alpha = \beta + 1$ for some β, then $\beta \in B$ since $\beta < \alpha$, implying $\alpha \in B$, contradicting the definition of α. Since $\alpha \in \epsilon_0$, we have

$$\alpha < \omega^\alpha = \sup_{\iota < \alpha} \omega^\iota$$

So, there exists $\beta < \alpha$ such that

$$\omega^\beta \leqslant \alpha < \omega^{\beta+1} = \omega^\beta \cdot \omega = \sup_{n < \omega} (\omega^\beta \cdot n)$$

Then, we can find $k \in \mathbf{N}$ and $\gamma < \omega^\beta$ such that

$$\alpha = \omega^\beta \cdot k + \gamma$$

Since $\omega^\beta, k, \gamma < \alpha$, these are all in B and so is α, contradicting the assumption. □

4.2 A system up to the Feferman–Schütte ordinal

In order to reach much larger ordinals, we consider computing the binary Veblen function [48]. We will first recall the construction of the binary Veblen function, and then implement it with a minor change in the framework of algorithmic term rewriting systems.

For the sake of completeness we briefly recapitulate some well-known basic notions about ordinals. We remark that the word 'nice' here is not a common terminology.

4.2.1 The binary Veblen function

First of all, we introduce the notion of *nice functions* as candidates of base functions of binary Veblen functions, and also the notions of *nice sets* and *nice functions*.*

Definition 147 Let A be a subset of $\mathbf{\Omega}$.

1. The set A is *unbounded* if $\forall \alpha \in \mathbf{\Omega}.\ \exists \beta \in A.\ \beta > \alpha$.
2. The set A is *closed* if $\sup B \in A$ for every countable $B \subseteq A$.
3. The set A is *nice* if A is unbounded and closed, and $0 \notin A$. ◹

Definition 148 A function $f : \mathbf{\Omega} \to \mathbf{\Omega}$ is *nice* if the following conditions are all satisfied.

1. The function f is $\mathbf{\Omega}$-*continuous*, i.e.
$$\sup_{\alpha \in A} f(\alpha) = f(\sup A)$$
for every countable $A \subseteq \mathbf{\Omega}$.
2. The function f is *strongly monotonic*, i.e. if $\alpha < \beta$, then $f(\alpha) < f(\beta)$.
3. Zero is not a fixed point of the function, i.e. $f(0) > 0$.

◹

Lemma 149 *Let f be a nice function. Then, we have $\alpha \leqslant f(\alpha)$ for all $\alpha \in \mathbf{\Omega}$.*

Proof: Let $X = \{\iota \mid f(\iota) < \iota\}$. For a proof by contradiction assume that $X \neq \varnothing$. Let $\alpha = \min X$. Obviously $\alpha > 0$.

If α is a successor ordinal with $\alpha = \beta + 1$, then from minimality of α we have $f(\beta) \geqslant \beta$. From strict monotonicity of f, we have $f(\alpha) > f(\beta)$ and thus $f(\alpha) \geqslant f(\beta) + 1 \geqslant \beta + 1 = \alpha$, contradicting that $\alpha \in X$.

*The word 'nice' appears also in [10] on their tree-ordinal notation, however our niceness is different from that.

If α is a limit ordinal, then from minimality of α we have $f(\beta) \geqslant \beta$ for all $\beta < \alpha$. Therefore, from $\mathbf{\Omega}$-continuity of f, we have

$$f(\alpha) = \sup_{\beta<\alpha} f(\beta) \geqslant \sup_{\beta<\alpha} \beta = \alpha$$

contradicting that $\alpha \in X$. □

Example 150 The function f given by

$$f(\alpha) = \omega^{\alpha}$$

is nice. On the other hand, the function g given by

$$g(\alpha) = \alpha^{\omega}$$

is not; we have $g(0) = 0$. Moreover, $\mathbf{\Omega}$-continuity fails; the equation

$$\sup_{\alpha\in A} g(\alpha) = g(\sup A)$$

does not hold for $A = \omega(= \mathbf{N})$. The lefthand-side is ω while the righthand-side is ω^{ω} (note that $n^{\omega} = \omega$ for $n < \omega$). ⊡

Definition 151 let A be an unbounded subset of $\mathbf{\Omega}$. Then, the *enumeration function*, written $\mathrm{enum}(A)$, is defined as the strongly monotonic function satisfying $\mathrm{img}(\mathrm{enum}(A)) = A$. Formally, $\mathrm{enum}(A)$ is given by transfinite induction as

$$(\mathrm{enum}(A))(\alpha) = \min(A \setminus \{(\mathrm{enum}(A))(\iota) \mid \iota < \alpha\}).$$

◪

Lemma 152 *A nice function and a nice set induce each other, i.e.*

1. *If* f *is a nice function, then* $\mathrm{img}(f)$ *is a nice set.*
2. *If* A *is a nice set, then* $\mathrm{enum}(A)$ *is a nice function.*

Proof:

1. (Unbounded) Immediately follows from Lemma 149.

   (Closed) Let B be a countable subset of $\mathrm{img}(f)$. Then, for each $\beta \in B$ we can find some $\alpha \in \mathbf{\Omega}$ such that $f(\alpha) = \beta$. Let

   $$A = \{\min\{\alpha \in \mathbf{\Omega} \mid f(\alpha) = \beta\} \mid \beta \in B\}$$

   Then, we have $\{f(\alpha) \mid \alpha \in A\} = B$ and thus

   $$f(\sup A) = \sup_{\alpha\in A} f(\alpha) = \sup B$$

from $\mathbf{\Omega}$-continuity of A. Therefore $\sup B \in \mathrm{img}(f)$. Hence, $\mathrm{img}(f)$ is closed.

(No-zero) Let $\alpha \in \mathbf{\Omega}$. We have

$$f(\alpha) = f(0 \vee \alpha) = f(0) \vee f(\alpha) \geqslant f(0) > 0$$

Hence, $0 \notin \mathrm{img}(f)$.

2. (Strongly monotonic) Follows easily from Definition 151.

   ($\mathbf{\Omega}$-continuous) Let B be a countable subset of $\mathbf{\Omega}$, and let

   $$C = \{(\mathrm{enum}(A))(\alpha) \mid \alpha \in B\}$$

   Obviously, C is a countable subset of A. From continuity of the nice set A, we have $\sup C \in A$.

   If $(\mathrm{enum}(A))(\sup B) < \sup C$, then there exists $\beta \in B$ such that $(\mathrm{enum}(A))(\sup B) < (\mathrm{enum}(A))(\beta)$. Since $\mathrm{enum}(A)$ is strongly monotonic, we have $\sup B < \beta \in B$, which is a contradiction. Thus, $(\mathrm{enum}(A))(\sup B) \geqslant \sup C$.

   Now, assume $(\mathrm{enum}(A))(\sup B) > \sup C$. Since $\sup C \in A$, we can find $\beta \in \mathbf{\Omega}$ such that $(\mathrm{enum}(A))(\beta) = \sup C$. By the assumption, we have $\sup B > \beta$. Thus, there exist some $\gamma \in B$ such that $\beta < \gamma$, implying $\sup C = (\mathrm{enum}(A))(\beta) < (\mathrm{enum}(A))(\gamma) \in C$, which is a contradiction. Thus, $(\mathrm{enum}(A))(\sup B) \leqslant \sup C$.

   Therefore, $(\mathrm{enum}(A))(\sup B) = \sup C$. Hence, $\mathrm{enum}(A)$ is $\mathbf{\Omega}$-continuous.

   (Non-Zero) Trivial. □

Definition 153 Given a function $f : \mathbf{\Omega} \to \mathbf{\Omega}$, the set of *fixed points* of f, written $\mathfrak{F}(f)$, is defined by

$$\mathfrak{F}(f) = \{\alpha \in \mathbf{\Omega} \mid f(\alpha) = \alpha\}.$$

◰

Lemma 154 *If f is a nice function, then so is* $\mathrm{enum}(\mathfrak{F}(f))$.

Proof: By Lemma 152, to show that $\mathfrak{F}(f)$ is nice will suffice.

(Unbounded) Given an ordinal α, let $\alpha_n = f^n(\alpha)$, formally,

$$\alpha_0 = \alpha$$
$$\alpha_{n+1} = f(\alpha_n).$$

Let $\beta = \sup_{n\in\mathbf{N}} \alpha_n$. Note that $\beta \geqslant \alpha$. By $\mathbf{\Omega}$-continuity of f,

$$\begin{aligned} f(\beta) &= \sup_{n\in\mathbf{N}} f(\alpha_n) && \text{(Since } f \text{ is } \mathbf{\Omega}\text{-continuous)} \\ &= \sup_{n\in\mathbf{N}} \alpha_{n+1} && \text{(By definition of } \alpha_n\text{)} \\ &= \sup_{m\in\mathbf{N}_+} \alpha_m && \text{(By letting } m = n+1\text{)} \\ &= \beta. && \text{(Since } \beta \geqslant \alpha = \alpha_0\text{)} \end{aligned}$$

Thus, $\beta \geqslant \alpha$ and $\beta \in \mathfrak{F}(f)$.

(Closed) Let A be a countable subset of $\mathfrak{F}(f)$. Then,

$$\begin{aligned} f(\sup A) &= \sup_{a\in A} f(a) && \text{(Since } f \text{ is } \mathbf{\Omega}\text{-continuous)} \\ &= \sup_{a\in A} a && \text{(By } A \subseteq \mathfrak{F}(f)\text{)} \\ &= \sup A. \end{aligned}$$

Therefore, $\sup A \in \mathfrak{F}(f)$.

(No-zero) Trivial. □

Given a nice function f, we write $f^{\Uparrow}$ to denote $\mathrm{enum}(\mathfrak{F}(f))$. That $f^{\Uparrow}$ grows much faster than the original function f. For example, let $f(\alpha) = \omega^{\alpha}$. Then, $f^{\Uparrow}(0)$ is already ϵ_0. In addition, given a nice set $A \subseteq \mathbf{\Omega}$, we can obtain a much sparser nice set $A^{\Uparrow}$ given as $\mathrm{img}(\mathrm{enum}(A)^{\Uparrow})$. It should be noticed that $A^{\Uparrow} \subseteq A$, since $A^{\Uparrow}$ consists of the ordinals α satisfying $(\mathrm{enum}(A))(\alpha) = \alpha$ and thus $\alpha \in A$.

Lemma 155 *Let λ be a limit ordinal, and let $\langle A_\iota\rangle_{\iota<\lambda}$ be a family of nice sets, indexed by λ. If $A_\iota \supseteq A_\kappa$ for all $\iota < \kappa < \lambda$, then $\bigcap_{\iota<\lambda} A_\iota$ also forms a nice set.*

Proof: Let $A_\lambda = \bigcap_{\iota<\lambda} A_\iota$. Moreover, since λ is countable, there exists a strongly increasing sequence $\langle\lambda_i\rangle_{i\in\mathbf{N}}$ leading to λ. Choose such a sequence as $\langle\lambda_i\rangle$.

(Unbounded) Let α be a limit ordinal. To show that $\exists\beta \geqslant \alpha.\ \beta \in A_\lambda$ will suffice. Let

$$\begin{aligned} \beta_0 &= \alpha \\ \beta_{n+1} &= \min\{\iota \in A_{\lambda_n} \mid \iota \geqslant \beta_n\}. \end{aligned}$$

Note that $m > n$ implies $\beta_m \in A_{\lambda_n}$. Now, let $\beta = \sup_{n\in\mathbf{N}} \beta_n$. Obviously, $\beta \geqslant \alpha$. And, from closedness of each A_{λ_n}, we have $\beta \in A_{\lambda_n}$ for every $n \in \mathbf{N}$ and thus $\beta \in A_\lambda$.

(Closed) Let B be a countable subset of A_λ. Then, we have $B \subseteq A_{\lambda_n}$ for every $n \in \mathbf{N}$. Thus, from closedness of each A_{λ_n}, we have $\sup B \in A_{\lambda_n}$. Hence, $\sup B \in A_{\lambda_n}$.

(No-Zero) Trivial. □

Definition 156 (The Veblen hierarchy) Given a nice set A, we define the *Veblen hierarchy generated by* A, written $\langle A^{(\iota)} \rangle_{\iota \in \mathbf{\Omega}}$, by transfinite induction as

$$A^{(0)} = A$$
$$A^{(\alpha+1)} = (A^{(\alpha)})^{\Uparrow}$$

and

$$A^{(\lambda)} = \bigcap_{\iota < \lambda} A^{(\iota)}$$

for a limit ordinal λ. Note that the hierarchy consists of nice sets, by Lemmas 152, 154, and 155.

Given a nice function $f : \mathbf{\Omega} \to \mathbf{\Omega}$, we define the *binary Veblen function generated by* f, denoted also by f but typed $\mathbf{\Omega} \times \mathbf{\Omega} \to \mathbf{\Omega}$, by

$$f(\alpha, \beta) = (\mathrm{enum}((\mathrm{img}(f))^{(\alpha)}))(\beta).$$

◪

In particular, we write ϕ to denote the binary Veblen function generated by $\phi(\alpha) = \omega^\alpha$. Table 25 shows a part of the function.

$\alpha \backslash \beta$	0	1	ω	ϵ_0	η_0	$\phi(\omega, 0)$
0	1	ω	ω^ω	ϵ_0	η_0	$\phi(\omega, 0)$
1	ϵ_0	ϵ_1	ϵ_ω	ϵ_{ϵ_0}	η_0	$\phi(\omega, 0)$
2	η_0	η_1	η_ω	η_{ϵ_0}	η_{η_0}	$\phi(\omega, 0)$
$\vdots$						
ω	$\phi(\omega, 0)$	$\phi(\omega, 1)$	$\phi(\omega, \omega)$	$\phi(\omega, \epsilon_0)$	$\phi(\omega, \eta_0)$	$\phi(\omega, \phi(\omega, 0))$

Table 25: $\phi(\alpha, \beta)$

4.2.2 Implementation

We wish to extend the system ORD-AME to devise ϕ as introduced in the previous subsection. However, it is difficult to implement ϕ exactly, because ϕ is not continuous in the first argument. Compare, for example, $\phi(\omega, \phi(\omega, 0))$ and $\sup_{\alpha < \omega} \phi(\alpha, \phi(\omega, 0))$. Since $\phi(\omega, 0)$ is defined as a common fixed point of $\phi(\alpha, -)$ up to $\alpha < \omega$, we have

$$\sup_{\alpha < \omega} \phi(\alpha, \phi(\omega, 0)) = \sup_{\alpha < \omega} \phi(\omega, 0) = \phi(\omega, 0)$$

while $\phi(\omega, \phi(\omega, 0)) > \phi(\omega, 0)$. Before we show the definitive version of the system, we see how this discontinuity prohibits an implementation of the function. The zeroth row can be easily obtained:

$$\phi(\mathsf{O}, n) \to \mathsf{exp}(\mathsf{omega}, n)$$

In order to compute following rows, we appeal to the following proposition:

Proposition 157 *Let f be a nice function. Then:*

1. *We have $\sup_{n\in\mathbf{N}} f^n(\alpha) = \min\{\iota \in \mathfrak{F}(f) \mid \iota \geqslant \alpha\}$, viz. the first fixed point of f that is greater than or equal to α.*
2. *We have $\sup_{n\in\mathbf{N}} f^n(\alpha+1) = \min\{\iota \in \mathfrak{F}(f) \mid \iota > \alpha\}$, viz. the first fixed point of f that is greater than α.*
3. *Given an infinite sequence $\langle\alpha_n\rangle_{n\in\mathbf{N}}$ such that $\alpha_i \in \mathfrak{F}(f)$ for every i, we have $\sup_{n\in\mathbf{N}} \alpha_n \in \mathfrak{F}(f)$.*

Proof: See the proof of Lemma 154 □

Thus, we introduce an auxiliary function symbol ξ with the rule

$$\xi(n, m) \to m : \xi(n, \phi(n, m))$$

which accumulates the operation $\phi(n, -)$, to obtain the sequence leading to the fixed point of $\phi(n, -)$ larger than or equal to m. Thereby, the rules

$$\begin{aligned}
\phi(\mathsf{S}\, n, \mathsf{O}) &\to \mathsf{L}(\xi(n, \mathsf{O})) \\
\phi(\mathsf{S}\, n, \mathsf{S}\, m) &\to \mathsf{L}(\xi(n, \mathsf{S}\, \phi(\mathsf{S}\, n, m))) \\
\phi(\mathsf{S}\, n, \mathsf{L}\, x) &\to \mathsf{L}(\phi^{\rightarrow}_{\mathsf{L}}(\mathsf{S}\, n, x)) \\
\phi^{\rightarrow}_{\mathsf{L}}(n, m : x) &\to \phi(n, m) : \phi^{\rightarrow}_{\mathsf{L}}(n, x)
\end{aligned}$$

follow. We have so far successfully implemented $\phi(\alpha, \beta)$ up to $\alpha < \omega$. But, it is hard to deal with 'limit rows'. One might notice

$$\begin{aligned}
\phi(\lambda, 0) &= \sup_{\alpha<\lambda} \phi(\alpha, 0) \\
\phi(\lambda, \beta+1) &= \sup_{\alpha<\lambda} \phi(\alpha, \phi(\lambda, \beta)+1)
\end{aligned}$$

and implement

$$\begin{aligned}
\phi(\mathsf{L}\, x, \mathsf{O}) &\to \mathsf{L}(\phi^{\leftarrow}_{\mathsf{L}}(x, 0)) \\
\phi(\mathsf{L}\, x, \mathsf{S}\, m) &\to \mathsf{L}(\phi^{\leftarrow}_{\mathsf{L}}(x, \mathsf{S}\, \phi(\mathsf{L}\, x, m))) \\
\phi(\mathsf{L}\, x, \mathsf{L}\, y) &\to \mathsf{L}(\phi^{\rightarrow}_{\mathsf{L}}(\mathsf{L}\, x, y)) \\
\phi^{\leftarrow}_{\mathsf{L}}(n : x, m) &\to \phi(n, m) : \phi^{\leftarrow}_{\mathsf{L}}(x, m)
\end{aligned}$$

using the auxiliary symbol $\phi^{\leftarrow}_{\mathsf{L}}$. At first sight, it seems to work. The problem is that $\mathsf{L}\,x$ does not always represent a limit ordinal. For example, the term $\mathsf{L}(\mathsf{O} : \mathsf{O} : \mathsf{O} : \dots)$ represents O. Thus, with the above rules, $\phi(0^{\omega}, 1)$ will be not correctly computed.

So, instead of ϕ, we implement a bi-continuous function Φ, which is defined by

$$\Phi(\alpha,\beta) = \begin{cases} \phi(\alpha,\beta) & (\text{if } \alpha < \omega) \\ \sup_{\iota<\alpha} \phi(\iota,\beta) & (\text{if } \alpha \text{ is a limit ordinal}) \\ \phi(\alpha',\beta). & (\text{if } \alpha \text{ is a successor ordinal larger than } \omega \\ & \qquad\qquad \text{with } \alpha = \alpha' + 1) \end{cases}$$

Table 26 shows a part of the function Φ. It should be noticed that

$$\Phi(\alpha+1,-) = (\Phi(\alpha,-))^{\Uparrow}$$

and $\Phi(\alpha,\beta) \geqslant \beta$ hold also for limit ordinals α. We have thus

$$\begin{aligned} \Phi(\alpha+1,0) &= \sup_{n\in\mathbf{N}} (\Phi(\alpha,-))^n(0) \\ \Phi(\alpha+1,\beta+1) &= \sup_{n\in\mathbf{N}} (\Phi(\alpha,-))^n(\Phi(\alpha+1,\beta)) \\ \Phi(\alpha+1,\sup B) &= \sup_{\beta\in B} \Phi(\alpha+1,\beta) \end{aligned}$$

so that the similar fixed point computation for ϕ will work also for Φ.

$\alpha\backslash\beta$	0	1	ω	ϵ_0	η_0	$\phi(\omega,0)$
0	1	ω	ω^{ω}	ϵ_0	η_0	$\phi(\omega,0)$
1	ϵ_0	ϵ_1	ϵ_{ω}	ϵ_{ϵ_0}	η_0	$\phi(\omega,0)$
2	η_0	η_1	η_{ω}	η_{ϵ_0}	η_{η_0}	$\phi(\omega,0)$
$\vdots$						
ω	$\phi(\omega,0)$	$\phi(\omega,0)$	$\phi(\omega,0)$	$\phi(\omega,0)$	$\phi(\omega,0)$	$\phi(\omega,0)$
$\omega+1$	$\phi(\omega,0)$	$\phi(\omega,1)$	$\phi(\omega,\omega)$	$\phi(\omega,\epsilon_0)$	$\phi(\omega,\eta_0)$	$\phi(\omega,\phi(\omega,0))$

Table 26: $\Phi(\alpha,\beta)$

Table 27 shows all the rules of the system $\mathsf{ORD}\text{-}\Gamma_0$. We accept the confusion of overloading between the function $\Phi : \mathbf{\Omega}\times\mathbf{\Omega}\to\mathbf{\Omega}$ and the defined function symbol $\Phi : \mathsf{ORD}\times\mathsf{ORD}\to\mathsf{ORD}$ which will expectedly compute the original Φ. Henceforth, we will similarly overload some signatures between functions and defined function symbols.

4.2.3 Productivity and representation limit

Proposition 158 *The system* $\mathsf{ORD}\text{-}\Gamma_0$ *is strongly productive.*

$$
\begin{aligned}
n + \mathsf{O} &\to n \\
n + \mathsf{S}\, m &\to \mathsf{S}(n + m) \\
n + \mathsf{L}\, x &\to \mathsf{L}(\mathsf{add}_{\mathsf{L}}(n, x)) \\
\mathsf{add}_{\mathsf{L}}(n, m : x) &\to (n + m) : \mathsf{add}_{\mathsf{L}}(n, x) \\
n \cdot \mathsf{O} &\to \mathsf{O} \\
n \cdot \mathsf{S}\, m &\to (n \cdot m) + n \\
n \cdot \mathsf{L}\, x &\to \mathsf{L}(\mathsf{mul}_{\mathsf{L}}(n, x)) \\
\mathsf{mul}_{\mathsf{L}}(n, m : x) &\to (n \cdot m) : \mathsf{mul}_{\mathsf{L}}(n, x) \\
\mathsf{exp}(\mathsf{O}, \mathsf{O}) &\to \mathsf{O} \\
\mathsf{exp}(\mathsf{S}\, n, \mathsf{O}) &\to \mathsf{S}\, \mathsf{O} \\
\mathsf{exp}(\mathsf{L}\, x, \mathsf{O}) &\to \mathsf{L}(\mathsf{pow0}_{\mathsf{L}}(x)) \\
\mathsf{exp}(n, \mathsf{S}\, m) &\to \mathsf{exp}(n, m) \cdot n \\
\mathsf{exp}(n, \mathsf{L}\, x) &\to \mathsf{L}(\mathsf{exp}_{\mathsf{L}}(n, x)) \\
\mathsf{pow0}_{\mathsf{L}}(n : x) &\to \mathsf{exp}(n, \mathsf{O}) : \mathsf{pow0}_{\mathsf{L}}(x) \\
\mathsf{exp}_{\mathsf{L}}(n, m : x) &\to \mathsf{exp}(n, m) : \mathsf{exp}_{\mathsf{L}}(n, x) \\
\mathsf{omega} &\to \mathsf{L}(\mathsf{nats}(\mathsf{O})) \\
\mathsf{nats}(n) &\to n : \mathsf{nats}(\mathsf{S}\, n) \\
\\
\Phi(\mathsf{O}, m) &\to \mathsf{exp}(\mathsf{omega}, m) \\
\Phi(\mathsf{S}\, n, \mathsf{O}) &\to \mathsf{L}(\Xi(n, \mathsf{O})) \\
\Phi(\mathsf{S}\, n, \mathsf{S}\, m) &\to \mathsf{L}(\Xi(n, \mathsf{S}(\Phi(\mathsf{S}\, n, m)))) \\
\Phi(\mathsf{S}\, n, \mathsf{L}\, y) &\to \mathsf{L}(\Phi^{\rightarrow}_{\mathsf{L}}(\mathsf{S}\, n, y)) \\
\Phi(\mathsf{L}\, x, m) &\to \mathsf{L}(\Phi^{\leftarrow}_{\mathsf{L}}(x, m)) \\
\Phi^{\rightarrow}_{\mathsf{L}}(n, m : y) &\to \Phi(n, m) : \Phi^{\rightarrow}_{\mathsf{L}}(n, y) \\
\Phi^{\leftarrow}_{\mathsf{L}}(n : x, m) &\to \Phi(n, m) : \Phi^{\leftarrow}_{\mathsf{L}}(x, m) \\
\Xi(n, m) &\to m : \Xi(n, \Phi(n, m))
\end{aligned}
$$

Table 27: *The system* ORD-Γ_0

Proof: (DN) Let ϕ' be the binary Veblen function generated by

$$\phi'(\alpha) = (\omega + 2)^{\alpha}$$

Then, the inductive height estimator $\mathbf{H}$ given by

$$\begin{aligned}
H^{+}(\alpha,\beta) = H^{\mathsf{add_L}}(\alpha,\beta) &= \alpha + \beta + 1 \\
H^{\cdot}(\alpha,\beta) = H^{\mathsf{mul_L}}(\alpha,\beta) &= (\alpha+1)(\beta+1) \\
H^{\mathsf{exp}}(\alpha,\beta) = H^{\mathsf{exp_L}}(\alpha,\beta) &= (2\cdot(\alpha+1))^{\beta+1} \\
H^{\mathsf{pow0_L}}(\alpha) &= 2\cdot(\alpha+1) \\
H^{\mathsf{omega}} &= \omega \\
H^{\mathsf{nats}}(\alpha) &= \alpha + \omega \\
H^{\Phi}(\alpha,\beta) = H^{\Phi^{\rightarrow}_{L}}(\alpha,\beta) = H^{\Phi^{\leftarrow}_{L}}(\alpha,\beta) &= \phi'(\alpha,\beta) \\
H^{\Xi}(\alpha,\beta) &= \min\{\iota \in \boldsymbol{\Omega} \mid \iota \geqslant \beta,\ \phi'(\alpha,\iota) = \iota\}
\end{aligned}$$

is adequate and strongly increasing. By Theorem 92, the system is DN.

(WN) With the inductive height estimator $\mathbf{H}$ as defined above, the root activity estimator $\mathbf{R}$ defined by

$$\begin{aligned}
R^{+}(\alpha,\beta) &= \alpha + 1 \\
R^{\cdot}(\alpha,\beta) &= (\alpha+1)\cdot\beta + 2 \\
R^{\mathsf{exp}}(\alpha,\beta) &= ((2\cdot(\alpha+1))^{\beta}+1)\cdot\alpha + 3 \\
R^{\Phi}(\alpha,\beta) &= ((\omega+2)^{\beta}+1)\cdot\omega + 4 \\
R^{\mathsf{add_L}}(\alpha,\beta) = R^{\mathsf{mul_L}}(\alpha,\beta) = R^{\mathsf{exp_L}}(\alpha,\beta) = R^{\mathsf{pow0_L}}(\alpha) = R^{\mathsf{omega}} &= \\
R^{\mathsf{nats}}(\alpha) = R^{\Phi^{\rightarrow}_{L}}(\alpha,\beta) = R^{\Phi^{\leftarrow}_{L}}(\alpha,\beta) = R^{\Xi}(\alpha,\beta) &= 1
\end{aligned}$$

is adequate. By Theorem 100, the system is WN.

(SCN) As we have dealt with the system ORD-AME in Examples 95 and 107, the system ORD-Γ_0 is also SCN, guaranteed by the adequate and strongly increasing inductive height estimator as given above.

Hence, by Corollary 41, the system ORD-Γ_0 is strongly productive. □

Remark 159 The adequate semantics on the system ORD-Γ_0 is given as follows:

$$\begin{aligned}
+^{\mathcal{A}}(\alpha,\beta) &= \alpha + \beta \\
\mathsf{add_L}^{\mathcal{A}}(\alpha,\boldsymbol{\beta}) &= \langle \alpha + \beta_i \rangle_{i\in\mathbf{N}} \\
\cdot^{\mathcal{A}}(\alpha,\beta) &= \alpha\cdot\beta \\
\mathsf{mul_L}^{\mathcal{A}}(\alpha,\boldsymbol{\beta}) &= \langle \alpha\cdot\beta_i \rangle_{i\in\mathbf{N}} \\
\mathsf{exp}^{\mathcal{A}}(\alpha,\beta) &= \alpha^{\beta} \\
\mathsf{pow0_L}^{\mathcal{A}}(\boldsymbol{\alpha}) &= \langle \alpha_i^{0} \rangle_{i\in\mathbf{N}}
\end{aligned}$$

$$\begin{aligned}
\mathsf{exp}_{\mathsf{L}}{}^{\mathcal{A}}(\alpha, \boldsymbol{\beta}) &= \langle \alpha^{\beta_i} \rangle_{i \in \mathbf{N}} \\
\mathsf{omega}^{\mathcal{A}}() &= \omega \\
\mathsf{nats}^{\mathcal{A}}(\alpha) &= \langle \alpha + i \rangle_{i \in \mathbf{N}} \\
\Phi^{\mathcal{A}}(\alpha, \beta) &= \Phi(\alpha, \beta) \\
\Phi_{\mathsf{L}}^{\rightarrow \mathcal{A}}(\alpha, \boldsymbol{\beta}) &= \langle \Phi(\alpha, \beta_i) \rangle_{i \in \mathbf{N}} \\
\Phi_{\mathsf{L}}^{\leftarrow \mathcal{A}}(\boldsymbol{\alpha}, \beta) &= \langle \Phi(\alpha_i, \beta) \rangle_{i \in \mathbf{N}} \\
\Xi^{\mathcal{A}}(\alpha, \beta) &= \langle (\Phi(n, -))^i(m) \rangle_{i \in \mathbf{N}}
\end{aligned}$$

Observe that this semantics is preserved under single-step reduction. Hence, by Lemma 78, it is adequate. ⧈

Proposition 160 *The system* ORD-Γ_0 *has the representation limit* Γ_0*, where* Γ_0 *is the Feferman–Schütte ordinal;* $\Gamma_0 = \min\{\alpha \in \boldsymbol{\Omega} \mid \phi(\alpha, 0) = \alpha\}$.

Proof: Similarly as the proof of Proposition 146. □

4.3 The small Veblen ordinal

One can extend the system ORD-Γ_0 by adding the defined symbols Γ, Γ_{L} and Δ with the rules

$$\begin{aligned}
\Gamma(\mathsf{O}) &\to \mathsf{L}(\Delta(\mathsf{O})) \\
\Gamma(\mathsf{S}\, n) &\to \Delta(\mathsf{S}\, \Gamma(n)) \\
\Gamma(\mathsf{L}\, x) &\to \mathsf{L}(\Gamma_{\mathsf{L}}(x)) \\
\Gamma_{\mathsf{L}}(n : x) &\to \Gamma(x) : \Gamma_{\mathsf{L}}(x) \\
\Delta(n) &\to n : \Delta(\Phi(n, \mathsf{O}))
\end{aligned}$$

to reach $[\![\Gamma(\mathsf{O})]\!] = \Gamma_0$. Observe that the representation limit of this system is $\min\{\alpha \in \boldsymbol{\Omega} \mid \Gamma_\alpha = \alpha\} = \Gamma_{\Gamma_{\Gamma_{\ldots}}}$, where

$$\Gamma_\alpha = (\mathrm{enum}\{\iota \in \boldsymbol{\Omega} \mid \phi(\iota, 0) = \iota\})(\alpha)$$

As to a systematic notation for such accumulations as above, accumulations of those accumulations, accumulations of accumulated accumulations, etc., we introduce the Veblen meta-hierarchy, which induces the function $\boldsymbol{\Omega}^* \to \boldsymbol{\Omega}$ from a nice function $\boldsymbol{\Omega} \to \boldsymbol{\Omega}$.

4.3.1 The Veblen meta-hierarchy

In order to go beyond Γ_0, we notice that $\phi(-, 0)$ is also a nice function. It holds for every binary Veblen function:

Lemma 161 *Let f be a binary Veblen function generated from a nice function f. Then, the unary function $f(-,0)$ forms a nice function.*

Proof: (Strongly monotonic) If $\alpha < \beta$, then from the definition of the binary Veblen function, we have $f(\alpha,0) < f(\alpha, f(\alpha,0)) \leqslant f(\beta,0)$.

($\boldsymbol{\Omega}$-continuous) Let λ be a limit ordinal, and let $\alpha = \sup_{\iota<\lambda} f(\iota,0)$ and $\beta = f(\lambda,0)$. To show that $\alpha = \beta$ will suffice. Since $f(-,0)$ is strongly monotonic as shown above, we have $\alpha \leqslant \beta$. On the other hand, for all $\gamma < \lambda$, we have

$$\begin{aligned} f(\gamma,\alpha) &= \sup_{\iota<\lambda} f(\gamma, f(\iota,0)) \\ &= \sup\{f(\gamma, f(\iota,0)) \mid \gamma < \iota < \lambda\} \\ &= \sup\{f(\iota,0) \mid \gamma < \iota < \lambda\} \\ &= \alpha \end{aligned}$$

and thus $\alpha \geqslant \beta$ from definition of the binary Veblen function. Hence, we have $\alpha = \beta$.

(Non-Zero) Trivial. □

Thus, given a nice function $f : \boldsymbol{\Omega} \to \boldsymbol{\Omega}$, we have *the first hierarchy* $f_0 : \boldsymbol{\Omega} \times \boldsymbol{\Omega} \to \boldsymbol{\Omega}$ as the binary Veblen function generated by f, *the second hierarchy* f_1 as the binary Veblen function generated by $f_0(-,0)$, and so on.[†] Moreover, for a limit ordinal λ, since $A_\lambda = \cap_{\alpha<\lambda}\{f_\alpha(\iota,0) \mid \iota \in \boldsymbol{\Omega}\}$ is a nice set by Lemma 155, we define the λ-th hierarchy $f_\lambda : \boldsymbol{\Omega} \times \boldsymbol{\Omega} \to \boldsymbol{\Omega}$ as the binary Veblen function generated by $\mathrm{enum}(A_\lambda)$. From this hierarchy of hierarchies, the *ternary* Veblen function arises as $f(\alpha,\beta,\gamma) = f_\alpha(\beta,\gamma)$. Furthermore, $\phi(-,0,0)$ is again a nice function. Thus, we can perform analogous procedures to construct the quaternary Veblen function, the quinary Veblen function, and so on.

Definition 162 Let f be a nice function. Then, the Veblen meta-hierarchy is given as the function $\mathbf{f} : \boldsymbol{\Omega}^* \to \boldsymbol{\Omega}$ defined as

$$\begin{aligned} \mathbf{f}(0,\ldots,0,\alpha_1,\ldots,\alpha_n) &= \mathbf{f}(\alpha_1,\ldots,\alpha_n) \\ \mathbf{f}(\alpha) &= f(\alpha) \\ \mathbf{f}(\alpha_1,\ldots,\alpha_n,\beta+1,\gamma) &= (\mathbf{f}(\alpha_1,\ldots,\alpha_n,\beta,-))^{\Uparrow}(\gamma) \\ \mathbf{f}(\alpha_1,\ldots,\alpha_n,\beta+1,\overbrace{0,\ldots,0}^{k+1},\gamma) &= \mathbf{f}(\alpha_1,\ldots,\alpha_n,\beta,\gamma,\overbrace{0,\ldots,0}^{k+1}) \end{aligned}$$

[†]Originally in [48], f_1 is defined as generated by $(f_0(-,0))^{\Uparrow}$, etc.. Thus, we have an extra row in each successor hierarchy.

and

$$\begin{aligned}&\mathbf{f}(\alpha_1,\ldots,\alpha_n,\lambda,\overbrace{0,\ldots,0}^{k},\gamma)\\&=\left(\operatorname{enum}\left(\bigcap_{\iota<\lambda}\{\mathbf{f}(\alpha_1,\ldots,\alpha_n,\iota,\overbrace{0,\ldots,0}^{k},\kappa)\mid\kappa\in\boldsymbol{\Omega}\}\right)\right)(\gamma)\end{aligned}$$

where λ is a limit ordinal. ◪

We write $\boldsymbol{\phi}$ to denote the Veblen meta-hierarchy generated by $\phi(\alpha)=\omega^\alpha$. For example, $\boldsymbol{\phi}(1,1,0)=\Gamma_0$ and $\boldsymbol{\phi}(1,2,0)=\Gamma_{\Gamma_{\Gamma_{\cdots}}}$.

4.3.2 Implementation

In order to implement the Veblen meta-hierarchy, we make a minor change of the meta-hierarchy so that

$$\boldsymbol{\Phi}(\sup A_1,\ldots,\sup A_n)=\sup\{\boldsymbol{\Phi}(\alpha_1,\ldots,\alpha_n)\mid\alpha_1\in A_1,\ldots,\alpha_n\in A_n\}$$

holds, similarly as Φ from ϕ. So, we define $\boldsymbol{\Phi}:\boldsymbol{\Omega}^*\to\boldsymbol{\Omega}$ by

$$\boldsymbol{\Phi}(0,\ldots,0,\alpha_1,\ldots,\alpha_n)=\boldsymbol{\Phi}(\alpha_1,\ldots,\alpha_n)\tag{a}$$
$$\boldsymbol{\Phi}(\alpha)=\omega^\alpha\tag{b}$$
$$\boldsymbol{\Phi}(\alpha_1,\ldots,\alpha_n,\beta+1,\gamma)=(\boldsymbol{\Phi}(\alpha_1,\ldots,\alpha_n,\beta+1,-))^{\Uparrow}(\gamma)\tag{c}$$
$$\boldsymbol{\Phi}(\alpha_1,\ldots,\alpha_n,\beta+1,\overbrace{0,\ldots,0}^{k+1},\gamma)=\boldsymbol{\Phi}(\alpha_1,\ldots,\alpha_n,\beta,\gamma,\overbrace{0,\ldots,0}^{k+1})\tag{d}$$
$$\boldsymbol{\Phi}(\alpha_1,\ldots,\alpha_n,\lambda,\overbrace{0,\ldots,0}^{k},\gamma)=\sup_{\iota<\lambda}\boldsymbol{\Phi}(\alpha_1,\ldots,\alpha_n,\iota,\overbrace{0,\ldots,0}^{k},\gamma).\tag{e}$$

Observe that we have

$$\boldsymbol{\phi}(\alpha_1,\ldots,\alpha_n,\beta,\overbrace{0,\ldots,0}^{k},\gamma)=\boldsymbol{\Phi}(\alpha_1,\ldots,\alpha_n,\beta',\overbrace{0,\ldots,0}^{k},\gamma')$$

where

$$\beta'=\begin{cases}\beta & (\text{if }\beta<\omega)\\ \beta+1 & (\text{if }\beta\geqslant\omega)\end{cases}\qquad\gamma'=\begin{cases}\gamma+1 & \left(\begin{array}{ll}\text{if} & \gamma\geqslant\omega\text{ and}\\ & (k>0\text{ or }\beta=0)\end{array}\right)\\ \gamma & (\text{otherwise})\end{cases}$$

similarly as in the binary Veblen function.

We employ the sort $\mathsf{LIST}_{\mathsf{ORD}}$ with

$$\mathsf{LIST}_{\mathsf{ORD}}\overset{\mathbf{I}}{:=}[\,]\mid\mathsf{ORD};\mathsf{LIST}_{\mathsf{ORD}}$$

$\alpha\backslash\beta$	0	1	2	3	$\cdots$	ω	$\omega+1$	$\cdots$	Γ_0	Γ_0+1	
0	1	ϵ_0	η_0	ζ_0	$\cdots$	$\phi(\omega,0)$	$\phi(\omega,0)$	$\cdots$	Γ_0	Γ_0	$\cdots$
1	Γ_0	Γ_1	Γ_2	Γ_3	$\cdots$	Γ_ω	$\Gamma_{\omega+1}$	$\cdots$	Γ_{Γ_0}	Γ_{Γ_0+1}	$\cdots$
2	$\Gamma_{\Gamma_{\Gamma_{\cdots}}}$										

Table 28: *The value of* $\mathbf{\Phi}(1,\alpha,\beta)$

for finite sequences (lists) of tree ordinals, and the defined symbol

$$\mathbf{\Phi} : \mathsf{LIST}_{\mathsf{ORD}} \to \mathsf{ORD}$$

to implement the function $\mathbf{\Phi} : \mathbf{\Omega}^* \to \mathbf{\Omega}$ with. We will construct the system such that $\mathbf{\Phi}([t_1;\ldots;t_n])$ computes $\mathbf{\Phi}([\![t_n]\!],\ldots,[\![t_1]\!])$.

Given a list of tree ordinals, we perform the following case analysis:

Case 1. The length of the list is less than 2. By Equations (a) and (b), we have the rules

$$\mathbf{\Phi}([\,]) \to \mathsf{S}\,\mathsf{O}$$
$$\mathbf{\Phi}(n;[\,]) \to \mathsf{exp}(\mathsf{omega},n).$$

Case 2. $\mathbf{\Phi}(n;\mathsf{O};l)$. In this case, for applying Equation (d), we have to count the number of leading zeros in l. To that purpose, we introduce an auxiliary defined symbol $\mathbf{\Upsilon} : \mathsf{ORD}\times\mathsf{NAT}\times\mathsf{LIST}_{\mathsf{ORD}} \to \mathsf{ORD}$. The rules

$$\mathbf{\Phi}(n;\mathsf{O};l) \to \mathbf{\Upsilon}(n,\mathsf{s}\,0,l)$$
$$\mathbf{\Upsilon}(n,k,\mathsf{O};l) \to \mathbf{\Upsilon}(n,\mathsf{s}\,k,l)$$

enable that counting. Then $\mathbf{\Upsilon}(n,k,l)$, where l has no leading zero, is left to be specified. If l is empty, then the whole list is of the form $n;0;\ldots;0$. Thus,

$$\mathbf{\Upsilon}(n,k,[\,]) \to \mathsf{exp}(\mathsf{omega},n)$$

similarly as $\mathbf{\Phi}(n;[\,])$.

If l begins with an S-type tree ordinal, then from Equation (d) the rule

$$\mathbf{\Upsilon}(n,k,\mathsf{S}\,m;l) \to \mathbf{\Phi}(\mathsf{insO}(k,n;m;l))$$

arises with insertion of zeros $\mathsf{insO} : \mathsf{NAT}\times\mathsf{LIST}_{\mathsf{ORD}} \to \mathsf{LIST}_{\mathsf{ORD}}$ specified by

$$\mathsf{insO}(0,l) \to l$$
$$\mathsf{insO}(\mathsf{s}\,k,l) \to \mathsf{insO}(k,\mathsf{O};l).$$

For l beginning with an L-type tree ordinal, the rules

$$\begin{aligned}\boldsymbol{\Upsilon}(n,k,\mathsf{L}\,x;l) &\to \mathsf{L}(\boldsymbol{\Upsilon}_{\mathsf{L}}(n,k,x,l))\\ \boldsymbol{\Upsilon}_{\mathsf{L}}(n,k,m:x,l) &\to \boldsymbol{\Upsilon}(n,k,m;l):\boldsymbol{\Upsilon}_{\mathsf{L}}(n,k,x,l)\end{aligned}$$

arise from Equation (e), similarly as exp with $\mathsf{exp}_{\mathsf{L}}$.

Case 3. $\boldsymbol{\Phi}(\mathsf{O};\mathsf{S}\,m;l)$. This part is very similar to the computation of $\Phi(\mathsf{S}\,n,m)$, the binary Veblen function. The tricky point is that $\boldsymbol{\Phi}(\ldots,0,-)$ is not even weakly increasing. For example, consider $\Phi(1,0,-)$ (see Table 28). The extra rows in the immediately preceding hierarchy causes that $\Phi(1,0,\lambda)=\Phi(1,0,\lambda+1)$ for every limit ordinal λ. In particular, when λ is a fixed point such as Γ_0, we have $\Phi(1,0,\Gamma_0+1)=\Phi(1,0,\Gamma_0)=\Gamma_0<\Gamma_0+1$. So, we take the second successor for the initial value of accumulation to construct next fixed points, as follows:

$$\begin{aligned}\boldsymbol{\Phi}(\mathsf{O};\mathsf{S}\,m;l) &\to \mathsf{L}(\boldsymbol{\Xi}(m;l,\mathsf{O}))\\ \boldsymbol{\Phi}(\mathsf{S}\,n;\mathsf{S}\,m;l) &\to \mathsf{L}(\boldsymbol{\Xi}(m;l,\mathsf{S}\,\mathsf{S}\,\boldsymbol{\Phi}(n;\mathsf{S}\,m;l)))\\ \boldsymbol{\Xi}(l,n) &\to n:\boldsymbol{\Xi}(l,\boldsymbol{\Phi}(n;l))\\ \boldsymbol{\Phi}(\mathsf{L}\,n;\mathsf{S}\,m;l) &\to \mathsf{L}(\boldsymbol{\Phi}^{\rightarrow}_{\mathsf{L}}(x,\mathsf{S}\,m;l))\\ \boldsymbol{\Phi}^{\rightarrow}_{\mathsf{L}}(n:x,l) &\to \boldsymbol{\Phi}(n;l):\boldsymbol{\Phi}^{\rightarrow}_{\mathsf{L}}(x,l)\end{aligned}$$

by Equation (c) and (e).

Case 4. $\boldsymbol{\Phi}(\mathsf{O};\mathsf{L}\,x;l)$. By Equation (e), we have the rules

$$\begin{aligned}\boldsymbol{\Phi}(n;\mathsf{L}\,x;l) &\to \mathsf{L}(\boldsymbol{\Phi}^{\leftarrow}_{\mathsf{L}}(n,x,l))\\ \boldsymbol{\Phi}^{\leftarrow}_{\mathsf{L}}(n,m:x,l) &\to \boldsymbol{\Phi}(n;m;l):\boldsymbol{\Phi}^{\leftarrow}_{\mathsf{L}}(n,x,l)).\end{aligned}$$

The whole system is presented in Table 29.

4.3.3 Productivity and representation limit

Proposition 163 *The system* ORD-SV *as in Table 29 is strongly productive.*

Proof: (DN and SCN) Let

$$\begin{aligned}\mathrm{H}^{+}(\alpha,\beta)=\mathrm{H}^{\mathsf{add}_{\mathsf{L}}}(\alpha,\beta)&=\alpha+\beta+1\\ \mathrm{H}^{\cdot}(\alpha,\beta)=\mathrm{H}^{\mathsf{mul}_{\mathsf{L}}}(\alpha,\beta)&=(\alpha+1)(\beta+1)\\ \mathrm{H}^{\mathsf{exp}}(\alpha,\beta)=\mathrm{H}^{\mathsf{exp}_{\mathsf{L}}}(\alpha,\beta)&=(2\cdot(\alpha+1))^{\beta+1}\\ \mathrm{H}^{\mathsf{pow0}_{\mathsf{L}}}(\alpha)&=2\cdot(\alpha+1)\\ \mathrm{H}^{\mathsf{omega}}&=\omega\end{aligned}$$

$$
\begin{aligned}
n + \mathsf{O} &\to n \\
n + \mathsf{S}\, m &\to \mathsf{S}(n + m) \\
n + \mathsf{L}\, x &\to \mathsf{L}(\mathsf{add}_{\mathsf{L}}(n, x)) \\
\mathsf{add}_{\mathsf{L}}(n, m : x) &\to (n + m) : \mathsf{add}_{\mathsf{L}}(n, x) \\
n \cdot \mathsf{O} &\to \mathsf{O} \\
n \cdot \mathsf{S}\, m &\to (n \cdot m) + n \\
n \cdot \mathsf{L}\, x &\to \mathsf{L}(\mathsf{mul}_{\mathsf{L}}(n, x)) \\
\mathsf{mul}_{\mathsf{L}}(n, m : x) &\to (n \cdot m) : \mathsf{mul}_{\mathsf{L}}(n, x) \\
\mathsf{exp}(\mathsf{O}, \mathsf{O}) &\to \mathsf{O} \\
\mathsf{exp}(\mathsf{S}\, n, \mathsf{O}) &\to \mathsf{S}\, \mathsf{O} \\
\mathsf{exp}(\mathsf{L}\, x, \mathsf{O}) &\to \mathsf{L}(\mathsf{pow0}_{\mathsf{L}}(x)) \\
\mathsf{exp}(n, \mathsf{S}\, m) &\to \mathsf{exp}(n, m) \cdot n \\
\mathsf{exp}(n, \mathsf{L}\, x) &\to \mathsf{L}(\mathsf{exp}_{\mathsf{L}}(n, x)) \\
\mathsf{pow0}_{\mathsf{L}}(n : x) &\to \mathsf{exp}(n, \mathsf{O}) : \mathsf{pow0}_{\mathsf{L}}(x) \\
\mathsf{exp}_{\mathsf{L}}(n, m : x) &\to \mathsf{exp}(n, m) : \mathsf{exp}_{\mathsf{L}}(n, x) \\
\mathsf{omega} &\to \mathsf{L}(\mathsf{nats}(\mathsf{O})) \\
\mathsf{nats}(n) &\to n : \mathsf{nats}(\mathsf{S}\, n) \\
\\
\boldsymbol{\Phi}([\,]) &\to \mathsf{S}\, \mathsf{O} \\
\boldsymbol{\Phi}(n; [\,]) &\to \mathsf{exp}(\mathsf{omega}, n) \\
\boldsymbol{\Phi}(n; \mathsf{O}; l) &\to \boldsymbol{\Upsilon}(n, \mathsf{s}\, 0, l) \\
\boldsymbol{\Phi}(\mathsf{O}; \mathsf{S}\, m; l) &\to \mathsf{L}(\boldsymbol{\Xi}(m; l, \mathsf{O})) \\
\boldsymbol{\Phi}(\mathsf{S}\, n; \mathsf{S}\, m; l) &\to \mathsf{L}(\boldsymbol{\Xi}(m; l, \mathsf{S}\, \mathsf{S}\, \boldsymbol{\Phi}(n; \mathsf{S}\, m; l))) \\
\boldsymbol{\Phi}(\mathsf{L}\, n; \mathsf{S}\, m; l) &\to \mathsf{L}(\boldsymbol{\Phi}^{\rightarrow}_{\mathsf{L}}(x, \mathsf{S}\, m; l)) \\
\boldsymbol{\Phi}(n; \mathsf{L}\, x; l) &\to \mathsf{L}(\boldsymbol{\Phi}^{\leftarrow}_{\mathsf{L}}(n, x, l)) \\
\boldsymbol{\Phi}^{\rightarrow}_{\mathsf{L}}(n : x, l) &\to \boldsymbol{\Phi}(n; l) : \boldsymbol{\Phi}^{\rightarrow}_{\mathsf{L}}(x, l) \\
\boldsymbol{\Phi}^{\leftarrow}_{\mathsf{L}}(n, m : x, l) &\to \boldsymbol{\Phi}(n; m; l) : \boldsymbol{\Phi}^{\leftarrow}_{\mathsf{L}}(n, x, l)) \\
\boldsymbol{\Xi}(l, n) &\to n : \boldsymbol{\Xi}(l, \boldsymbol{\Phi}(n; l)) \\
\boldsymbol{\Upsilon}(n, k, \mathsf{O}; l) &\to \boldsymbol{\Upsilon}(n, \mathsf{s}\, k, l) \\
\boldsymbol{\Upsilon}(n, k, [\,]) &\to \mathsf{exp}(\mathsf{omega}, n) \\
\boldsymbol{\Upsilon}(n, k, \mathsf{S}\, m; l) &\to \boldsymbol{\Phi}(\mathsf{insO}(k, n; m; l)) \\
\boldsymbol{\Upsilon}(n, k, \mathsf{L}\, x; l) &\to \mathsf{L}(\boldsymbol{\Upsilon}_{\mathsf{L}}(n, k, x, l)) \\
\boldsymbol{\Upsilon}_{\mathsf{L}}(n, k, m : x, l) &\to \boldsymbol{\Upsilon}(n, k, m; l) : \boldsymbol{\Upsilon}_{\mathsf{L}}(n, k, x, l) \\
\mathsf{insO}(0, l) &\to l \\
\mathsf{insO}(\mathsf{s}\, k, l) &\to \mathsf{insO}(k, \mathsf{O}; l)
\end{aligned}
$$

Table 29: *The system* ORD-SV

$$\begin{aligned}
\mathrm{H}^{\mathsf{nats}}(\alpha) &= \alpha + \omega \\
\mathrm{H}^{\boldsymbol{\Phi}}(\alpha) &= \boldsymbol{V}(\alpha) \\
\mathrm{H}^{\boldsymbol{\Phi}^{\rightarrow}_{\mathsf{L}}}(\alpha,\beta) &= \boldsymbol{V}(\alpha \vee (\beta+1)) \\
\mathrm{H}^{\boldsymbol{\Phi}^{\leftarrow}_{\mathsf{L}}}(\alpha,\beta,\gamma) &= \boldsymbol{V}(\alpha \vee (\beta+1) \vee (\gamma+2)) \\
\mathrm{H}^{\Xi}(\alpha,\beta) &= \boldsymbol{V}(\alpha \vee \beta) \\
\mathrm{H}^{\Upsilon}(\alpha,\beta,\gamma) &= \boldsymbol{V}(\alpha \vee (\gamma+\beta+1)) \\
\mathrm{H}^{\Upsilon_{\mathsf{L}}}(\alpha,\beta,\gamma,\delta) &= \boldsymbol{V}(\alpha \vee (\gamma+\beta+1) \vee (\delta+1+\beta+1)) \\
\mathrm{H}^{\mathsf{insO}}(\alpha,\beta) &= \beta + \alpha
\end{aligned}$$

where $\boldsymbol{V} : \boldsymbol{\Omega} \to \boldsymbol{\Omega}$ is defined by

$$\boldsymbol{V}(\alpha) = \sup\{\boldsymbol{\phi}(\alpha)+1, \boldsymbol{\phi}(\alpha,\alpha)+1, \boldsymbol{\phi}(\alpha,\alpha,\alpha)+1, \dots\}.$$

Observe that the induced inductive height estimator is adequate and strongly increasing.

(WN) Unfortunately, there exists no adequate root activity estimator with the above inductive height estimator (see Remark 165). So, we will construct another interpretation based on the expected semantics, derived from the following continuous Σ-algebra $\langle \boldsymbol{\Omega}^{\omega}, [-] \rangle$, which is weakly monotonic.

$$\begin{aligned}
[-]^{\mathsf{O}}() &= 0 \\
[-]^{\mathsf{S}}(\alpha) &= \alpha + 1 \\
[-]^{\mathsf{L}}(\alpha) &= \alpha \\
[-]^{\cdot}(\alpha,\beta) &= \sup\{\alpha,\beta\} \\
[-]^{[]}() &= 0 \\
[-]^{;}(\alpha,\beta) &= \boldsymbol{\Omega}\cdot\beta + \alpha \\
[-]^{+}(\alpha,\beta) &= \alpha + \beta \\
[-]^{\cdot}(\alpha,\beta) = [-]^{\mathsf{mul}_{\mathsf{L}}} &= \alpha \cdot \beta \\
[-]^{\mathsf{exp}}(\alpha,\beta) = [-]^{\mathsf{exp}}_{\mathsf{L}}(\alpha,\beta) &= \alpha^{\beta} \\
[-]^{\mathsf{pow0}}_{\mathsf{L}}(\alpha) &= 0^{\alpha} \\
[-]^{\omega}() &= \omega \\
[-]^{\mathsf{nats}}(\alpha) &= \alpha + \omega \\
[-]^{\boldsymbol{\Phi}}(\boldsymbol{\Omega}^{n}\cdot\alpha_n + \cdots + \boldsymbol{\Omega}\cdot\alpha_1 + \alpha_0) &= \Phi(\alpha_n,\dots,\alpha_1,\alpha_0) \\
[-]^{\boldsymbol{\Phi}^{\rightarrow}_{\mathsf{L}}}(\alpha,\beta) &= [-]^{\boldsymbol{\Phi}}(\boldsymbol{\Omega}\cdot\beta+\alpha) \\
[-]^{\boldsymbol{\Phi}^{\leftarrow}_{\mathsf{L}}}(\alpha,\beta,\gamma) &= [-]^{\boldsymbol{\Phi}}(\boldsymbol{\Omega}^2\cdot\gamma + \boldsymbol{\Omega}\cdot\beta+\alpha) \\
[-]^{\Xi}(\alpha,\beta) &= \min\{\iota \mid \iota \geqslant \beta, [-]^{\boldsymbol{\Phi}}(\boldsymbol{\Omega}\cdot\beta+\iota) = \iota\} \\
[-]^{\Upsilon}(\alpha,\beta,\gamma) &= [-]^{\boldsymbol{\Phi}}(\boldsymbol{\Omega}^{\beta+1}\cdot\gamma+\alpha)
\end{aligned}$$

$$
\begin{aligned}
[-]^{\Upsilon_{\mathsf{L}}}(\alpha,\beta,\gamma,\delta) &= [-]^{\Upsilon}(\alpha,\beta,\boldsymbol{\Omega}\cdot\delta+\gamma)\\
[-]^{0}() &= 0\\
[-]^{\mathsf{S}}(\alpha) &= \alpha+1\\
[-]^{\mathsf{insO}}(\alpha,\beta) &= \boldsymbol{\Omega}^{\alpha}+\beta
\end{aligned}
$$

where we stipulate that $0^0 = 0$. Note that

$$
[t] < \begin{cases} \omega & (\text{if } \mathrm{srt}(t) = \mathsf{NAT})\\ \boldsymbol{\Omega} & (\text{if } \mathrm{srt}(t) = \mathsf{ORD}, \mathsf{STREAM}_{\mathsf{ORD}})\\ \boldsymbol{\Omega}^{\omega} & (\text{if } \mathrm{srt}(t) = \mathsf{LIST}_{\mathsf{ORD}}) \end{cases}
$$

for well-definedness. The induced interpretation $[-]^{\zeta} : \mathcal{T}^{\star} \to \boldsymbol{\Omega}^{\omega}$ satisfies $[l]^{\zeta} \geqslant [r]^{\zeta}$ for all $(l, r) \in \mathcal{R}$, for every variable interpretation $\zeta : \mathcal{X} \to \boldsymbol{\Omega}^{\omega}$ such that

$$
[x] < \begin{cases} \omega & (\text{if } \mathrm{srt}(x) = \mathsf{NAT})\\ \boldsymbol{\Omega} & (\text{if } \mathrm{srt}(x) = \mathsf{ORD}, \mathsf{STREAM}_{\mathsf{ORD}})\\ \boldsymbol{\Omega}^{\omega}. & (\text{if } \mathrm{srt}(x) = \mathsf{LIST}_{\mathsf{ORD}}) \end{cases}
$$

Thus, $t \to s$ implies $[t] \geqslant [s]$.

Now, let a function $\mathsf{R}^{f} : \boldsymbol{\Omega}^{\omega} \times \cdots \times \boldsymbol{\Omega}^{\omega} \to \boldsymbol{\Omega}^{\omega}$ for each $f \in \Sigma$ be given as follows.

$$
\begin{aligned}
\mathsf{R}^{+}(\alpha,\beta) &= \alpha+1\\
\mathsf{R}^{\cdot}(\alpha,\beta) &= \alpha\cdot\beta+1\\
\mathsf{R}^{\mathsf{exp}}(\alpha,\beta) &= \alpha^{\beta}+1\\
\mathsf{R}^{\boldsymbol{\Phi}}(\alpha) &= \omega^{\alpha}+\omega\\
\mathsf{R}^{\Upsilon}(\alpha,\beta,\gamma) &= \omega^{\boldsymbol{\Omega}^{\beta+1}\cdot\gamma+\alpha} + \min\{\iota \mid \gamma < \boldsymbol{\Omega}^{\iota}\} + 2\\
\mathsf{R}^{\mathsf{insO}}(\alpha,\beta) &= \boldsymbol{\Omega}^{\alpha+1}\cdot\beta+\alpha+1\\
\mathsf{R}^{\mathsf{add}_{\mathsf{L}}}(\alpha,\beta) = \mathsf{R}^{\mathsf{mul}_{\mathsf{L}}}(\alpha,\beta) &= \mathsf{R}^{\mathsf{pow0}_{\mathsf{L}}}(\alpha) = \mathsf{R}^{\mathsf{exp}_{\mathsf{L}}}(\alpha,\beta) = \mathsf{R}^{\mathsf{omega}} =\\
\mathsf{R}^{\mathsf{nats}}(\alpha) = \mathsf{R}^{\boldsymbol{\Phi}^{\rightarrow}_{\mathsf{L}}}(\alpha,\beta) &= \mathsf{R}^{\boldsymbol{\Phi}^{\leftarrow}_{\mathsf{L}}}(\alpha,\beta,\gamma) = \mathsf{R}^{\Xi}(\alpha,\beta) = \mathsf{R}^{\Upsilon_{\mathsf{L}}}(\alpha,\beta,\gamma,\delta) = 1
\end{aligned}
$$

These functions with $[-]$ induce a $\to_{\epsilon} | \to$-compatible well-founded quasiorder on $\mathcal{G}^{\star}$ as well as an adequate root activity estimator with an adequate inductive height estimator does. □

Remark 164 The adequate semantics on the system ORD-SV is given exactly as we intended or sketched, whose adequacy can be ensured similarly as in Remark 159. ⧈

Clearly, $\mathbf{V}(1)$ as defined in the DN part of the proof of Proposition 163 is the representation limit of this system. This ordinal is called the *small Veblen ordinal*, which is sometimes written as $\theta_{\boldsymbol{\Omega}^{\omega}}(0)$. The definition of θ originates from Miller's $\boldsymbol{\omega}$ [36] and is comprehensively explained by Levitz [35].

Remark 165 Here we show that there exists no adequate root activity estimator with the adequate inductive height estimator $\mathbf{H}$ as defined in the above proof. Assume that there exists an adequate root activity estimator $\mathbf{R}$. Then, from the rule

$$\mathbf{\Phi}(n; \mathsf{O}; l) \to \mathbf{\Upsilon}(n, \mathsf{s}\,0, l)$$

we have

$$\mathrm{R}^{\mathbf{\Phi}}(\sup\{n+1, l+2\}) > \mathrm{R}^{\mathbf{\Upsilon}}(n, 1, l)$$

and thus

$$\mathrm{R}^{\mathbf{\Phi}}(4) > \mathrm{R}^{\mathbf{\Upsilon}}(0, 1, 2)$$

by letting $(n, l) = (0, 2)$. On the other hand, from the rule

$$\mathbf{\Upsilon}(n, k, \mathsf{S}\,m; l) \to \mathbf{\Phi}(\mathsf{insO}(k, n; m; l))$$

we have

$$\begin{aligned}\mathrm{R}^{\mathbf{\Upsilon}}(n, k, \sup\{m+2, l+1\}) &> \mathrm{R}^{\mathbf{\Phi}}(\mathrm{H}^{\mathsf{insO}}(k, \sup\{n+1, m+2, l+2\})) \\ &= \mathrm{R}^{\mathbf{\Phi}}(\sup\{n+1, m+2, l+2\} + k)\end{aligned}$$

and thus

$$\mathrm{R}^{\mathbf{\Upsilon}}(0, 1, 2) > \mathrm{R}^{\mathbf{\Phi}}(4)$$

by letting $(n, k, m, l) = (0, 1, 0, 1)$. ⧈

Remark 166 The Feferman–Schütte ordinal Γ_0 is known as the smallest impredicative ordinal [20], though the definition of the word 'predicative' is controversial (cf. [52]). Even if we admit that Γ_0 is impredicative, this does not contradict that our system covers Γ_0, because the system 'predicatively *defines*' the ordinal Γ_0 but does not 'predicatively *prove*' that the set Γ_0 is well-ordered [51]. In fact, every recursive ordinal is predicatively *definable* [50]. So, the author conjectures that for every ordinal λ which is less than the Church–Kleene ordinal ω_1^{CK}, there exists a system of tree ordinals with its representation limit greater than λ. ⧈

4.4 Hydra games

To conclude this chapter, we briefly point out the relation between tree ordinals and Hydra games.

Hydra games originate from Kirby and Paris [31]. Roughly speaking, a hydra is an unlabeled finitely branching tree. The reduction relation between hydrae induces a partial order of the order type ϵ_0. In [26] we established that hydrae are closely related to the system ORD-AME.

Buchholz has introduced an extended notion of hydra games in [6]. There, each non-root node of a hydra is labeled with $\overline{\mathbf{N}}$. The original hydra

games can be then recognized as hydrae with all the non-root nodes labeled 0. The order type of the partial order given by the reduction on Buchholz's hydra games is Γ_0 [53]. However, as far as I know, there is no clear relation between Buchholz's hydra games and the system ORD-Γ_0.

As to further extensions, one may think of labeling nodes of a hydra by hydrae themselves. I have not encountered this extension in the literature, and am curious which ordinal corresponds to this class of 'superhydrae'. However, this question is beyond the scope of the present work.

Bibliography

[1] A. Abel. Mixed inductive/coinductive types and strong normalization. In *APLAS 2007*, pp. 286–301, 2007.

[2] H.P. Barendegt. *The Lambda Calculus*. North-Holland, 1987.

[3] P. Bernays. *Axiomatic Set Theory*. Dover Publications, 1991.

[4] Y. Bertot, E. Komendantskaya. Using structual recursion for corecursion. In *TYPES*, pp. 220–236, 2008.

[5] I. Bethke, J. W. Klop, R. C. de Vrijer. Descendants and origins in term rewriting. *Information and Computation*, 159(1–2):59–124, 2000.

[6] W. Buchholz. An independence result for $(\Pi^1_1\text{-CA}) + \text{BI}$. *Annals of Pure and Applied Logic*, 33(2):131–155, 1987.

[7] W. Buchholz. A term calculus for (co-)recursive definitions on streamlike data structures. *Annals of Pure and Applied Logic*, 136(1-2):75–90, 2005.

[8] J.H. Conway. *On Numbers and Games, Second Edition*. A K Peters Ltd., 2000.

[9] Th. Coquand. Infinite Objects in Type Theory. In H. Barendregt, T. Nipkow, eds., *TYPES*, vol. 806, pp. 62–78. Springer-Verlag, Berlin, 1994.

[10] E.C. Dennis-Jones, S.S.Wainer. Subrecursive hierarchies via directed limits. *Lecture Notes in Mathematics*, 1104:117–128, 1983.

[11] N. Dershowitz. Orderings for term rewriting systems. *Theoretical Computer Science*, 17(3):279–301, 1982.

[12] N. Dershowitz. Trees, ordinals and termination. In *TAPSOFT*, pp. 243–250, 1993.

[13] N. Dershowitz, E. M. Reingold. Ordinal arithmetic with list structures. In *LFCS*, pp. 117–126, 1992.

[14] E.W. Dijkstra. On the productivity of recursive definitions, 1980. EWD749.

[15] J. Endrullis. Personal communication.

[16] J. Endrullis, C. Grabmayer, D. Hendriks. Data-oblivious stream productivity. In *LPAR*, pp. 79–96, 2008.

[17] J. Endrullis, C. Grabmayer, D. Hendriks, A. Isihara, J.W. Klop. Productivity of stream definitions. In *FCT 2007*, LNCS 4639, pp. 274–287, 2007.

[18] J. Endrullis, C. Grabmayer, D. Hendriks, A. Isihara, J.W. Klop. Productivity of stream definitions. *Theoretical Computer Science*, 411:765–782, 2010.

[19] J. Endrullis, C. Grabmayer, D. Hendriks, J. W. Klop, R. C. de Vrijer. Proving infinitary normalization. In *TYPES 2008*, pp. 64–82, 2008.

[20] S. Feferman. Predicativity. In *The Oxford Handbook of Philosophy of Mathematics and Logic*, pp. 590–624. Oxford University Press, 2005.

[21] J. Gallier. What's so special about Kruskal's theorem and the ordinal Γ_0? A survey of some results in proof theory. *Annals of Pure and Applied Logic*, 53:199–260, 1991.

[22] P. Di Gianantonio, M. Miculan. A unifying approach to recursive and co-recursive definitions. In *TYPES*, pp. 148–161, 2002.

[23] J. A. Goguen, J. W. Thatcher, E. G. Wagner, J. B. Wright. Initial algebra semantics and continuous algebras. *Journal of the Association for Computing Machinery*, 24(1):68–95, 1977.

[24] P. Halmos. How to write mathematics. *L'Enseignement Mathématique*, 16:123–152, 1970.

[25] J. Hughes, L. Pareto, A. Sabry. Proving the correctness of reactive systems using sized types. In *POPL '96*, pp. 410–423, 1996.

[26] A. Isihara. Hydra games and tree ordinals. In *WoLLIC 2007*, LNCS 4576, pp. 238–247, 2007.

[27] A. Isihara. Productivity of algorithmic systems. In *Proceedings of SCSS 2008*, number 08-08 in RISC-Linz Report, pp. 81–95, 2008.

[28] A. Isihara. Essay on transfinite terms. In J. W. Klop, V. van Oostrom, F. van Raamsdonk, eds., *Liber Amicorum for Roel de Vrijer*, pp. 109–116. Private publishing, 2009.

[29] R. Kennaway, F. de Vries. Infinitary rewriting. In Term Rewriting Systems [46], Chapter 12, pp. 668–711.

[30] R. Kennaway, J.W. Klop, M.R. Sleep, F. de Vries. Transfinite reductions in orthogonal term rewriting systems. *Information and Computation*, 119(1):18–38, 1995.

[31] L. Kirby, J. Paris. Accessible independence results for Peano Arithmetic. *Bulletin of the London Mathematical Society*, 14:285–293, 1982.

[32] J. W. Klop. New fixed point combinators from old. In E. Barendsen, H. Geuvers, V. Capretta, M. Niqui, eds., *Reflections on Type Theory, Lambda Calculus, and the Mind: Essays dedicated to Henk Barendregt on the occasion of his 60th birthday*, pp. 197–210. Radboud Universiteit Nijmegen, 2007.

[33] J.W. Klop, R.C. de Vrijer. Infinitary normalization. In S. Artemov, H. Barringer, A.S. d'Avila Garcez, L.C. Lamb, J. Woods, eds., *We Will Show Them: Essays in Honour of Dov Gabbay*, vol. 2, pp. 169–192. College Publications, 2005.

[34] J.W. Klop, V. van Oostrom, R.C. de Vrijer. Iterative lexicographic path orders. In *Essays Dedicated to Joseph A. Goguen*, LNCS 4060, pp. 541–554, 2006.

[35] H. Levitz. Transfinite ordinals and their notations: For the uninitiated, version 1.1. Currently available at `http://www.cs.fsu.edu/~levitz/research.html`.

[36] L. W. Miller. Normal functions and constructive ordinal notations. *The Journal of Symbolic Logic*, 41(2):439–459, 1976.

[37] M. Niqui. Productivity of Edalat-Potts exact arithmetic in constructive type theory. *Theory of Computing Systems*, 41(1):127–154, 2007.

[38] M. Niqui. Coalgebraic reasoning in Coq: bisimulation and the lambda-coiteration scheme. In *TYPES*, pp. 272–288, 2008.

[39] M. O. Rabin. Decidability of second order theories and automata on infinite trees. *Transactions of the American Mathematical Society*, 141:1–35, 1969.

[40] D. Schmidt. *Well-Partial Orderings and Their Maximal Order Types*. Habilitationsschrift, Ruprecht–Karl–Universität Heidelberg, 1979.

[41] K. Schütte. Kennzeichnung von Ordnungszahlen durch rekursiv erklärte Functionen. *Mathematische Annalen*, 127:15–32, 1954.

[42] K. Schütte. *Proof Theory*. Springer-Verlag, Berlin, 1977.

[43] W. Sierpiński. *Cardinal and Ordinal Numbers*. PWN Polish Scientific Publishers, 1965.

[44] B.A. Sijtsma. On the productivity of recursive list definitions. *ACM Transactions on Programming Languages and Systems*, 11(4):633–649, 1989.

[45] A. Telford, D. Turner. Ensuring the productivity of infinite structures. Technical Report 14-97, The Computing Laboratory, University of Kent at Canterbury, 1997.

[46] Terese. *Term Rewriting Systems*. Cambridge University Press, 2003.

[47] W. Thomas. Automata on infinite objects. In V. Leeuwen, ed., *Handbook of Theoretical Computer Science*, pp. 133–164. Elsevier, 1990.

[48] O. Veblen. Continuous increasing functions of finite and transfinite ordinals. *Transactions of the American Mathematical Society*, 9:280–292, 1908.

[49] W. W. Wadge. An extensional treatment of dataflow deadlock. *TCS*, 13:3–15, 1981.

[50] H. Wang. Ordinal numbers and predicative set theory. *Zeitschrift für mathematische Logik und Grundlagen der Mathematik*, 5:216–239, 1959.

[51] N. Weaver. Personal communication.

[52] N. Weaver. Predicativity beyond Gamma_0. Currently available at `http://arxiv.org/abs/math/0509244`.

[53] A. Weiermann. Personal communication.

[54] H. Zantema. Normalization of infinite terms. In *RTA*, pp. 441–455, 2008.

[55] Hans Zantema. Well-definedness of streams by termination. In *RTA*, pp. 164–178, 2009.

List of notations

$t\sigma$	substitution	15
$(\!\vert - \vert\!)_{\mathbf{R}}$	root activity estimation	68
$\wr u \int$	refinements of a pattern u	74
$\wr\!\wr \gamma \int\!\int$	instances of an observation γ	74
$\Vert p \Vert_t$	coinductive depth of a position p in a term t	52
$\trianglelefteqslant$	prefix	10
$\triangleleft$	proper prefix	11
$\blacktriangleright$	term concealing	74
$\searrow$	immediate subterm relation	43
$\searrow^{\mathcal{D}}$	— by functional application	43
$\searrow^{\mathbf{I}}$	— by inductive construction	43
$\searrow^{\mathbf{C}}$	— by coinductive construction	43
$\twoheadrightarrow$	finite reduction	17
$\twoheadrightarrow\!\!\!\!\twoheadrightarrow$	transfinite reduction	17
$\rightarrowtriangle$	requirement dependency relation	83
	ramified —	87
$\rightarrowtriangle^{N}$	nested —	83
	ramified nested —	88
$\rightarrowtriangle^{O}$	outermost —	83
	ramified outermost —	88
p/q	subsequence, $q \cdot (p/q) = p$	10
t/p	subterm of t at position p	15
$\bigvee$	maximum	11
$\bigwedge$	minimum	11
$\vert p \vert$	depth (length) of a position p	11
$\lceil - \rceil$	inductive height	43
$\lfloor - \rfloor_\gamma$	γ-quantity	78
$\Vert\!\lfloor - \rfloor\!\Vert_\gamma$	potential γ-quantity	78
$\lceil\!\lceil - \rceil\!\rceil_{\mathbf{H}}$	inductive height estimation	62
ar	arity, $(\Sigma \cup \mathcal{X}) \to \mathbf{N}$	14
$\mathcal{C}$	constructor function symbols	17
$\chi^{\mathbf{I}}, \chi^{\mathbf{C}}$	characteristic functions $\mathcal{S} \to \{0,1\}$	18
$\chi^{\mathbf{I}}_{\text{in}}, \chi^{\mathbf{C}}_{\text{in}}, \chi^{\mathbf{I}}_{\text{out}}, \chi^{\mathbf{C}}_{\text{out}}$	sugared notations (e.g. $\chi^{\mathbf{I}}_{\text{in}} = \chi^{\mathbf{I}} \circ \text{in}$)	18
cnf	(infinitary) constructor normal form	17
$\mathcal{D}$	defined function symbols	17

dom	domain	11
enum	enumeration function	96
ϵ	empty sequence	10
$\mathcal{F}$	finite terms	15
$\mathfrak{F}$	fixed points	97
$\mathcal{G}$	ground terms	15
$\mathcal{G}^\star$	proper ground terms	20
$\mathcal{G}^{\star\twoheadrightarrow}$	initially proper ground terms	21
Γ_0	Feferman–Schütte ordinal	104
img	image (range, codomain)	11
in	input sorts, $\Sigma \times \mathbf{N}_+ \rightharpoonup \mathcal{S}$	14
κ	canonical coinductive gauge	79
lbl	requirement labeling	80
$\mathbf{N}$	natural numbers	10
$\mathcal{N}$	normal forms	16
$\mathbf{N}_+$	positive integers	10
$\mathbf{N}_+^*$	positions in a term	11
$\mathcal{O}$	observations	74
$\mathbf{\Omega}$	countable ordinals	10
ω_1^{CK}	Church-Kleene ordinal	94
$\mathbf{On}$	ordinal numbers	10
out	output sort, $(\Sigma \cup \mathcal{X}) \to \mathcal{S}$	14
$\mathcal{P}$	constructor patterns	74
ra	root activity	67
$\text{rank}_{\preccurlyeq}$	rank according to the quasiorder $\preccurlyeq$	41
rdx	redex-requirement	80
$\boldsymbol{\rho}$	pre-requirement	80
ρ	γ-requirement function	79
$\mathcal{S}$	sorts	14
$\mathcal{S}^{\mathbf{C}}$	coinductive sorts	18
$\mathcal{S}^{\mathbf{I}}$	inductive sorts	18
Σ	function symbols	14
$\text{srt}(t)$	sort of a term t	14
$\mathcal{T}^\star$	proper terms	20
$\mathcal{T}^{\star\twoheadrightarrow}$	initially proper terms	21
trunC	coinductive truncation, $\mathcal{T} \times \mathbf{N} \to \mathcal{P}$	79
$\mathcal{V}$	constructor normal forms	17
$\mathcal{V}^\star$	proper constructor normal forms	20
$\mathcal{X}$	variable symbols	14
$t\{p \mapsto s\}$	replacing the subterm t/p by s	15

Index

Samenvatting

Algoritmische termherschrijfsystemen

In de theoretische informatica, in het bijzonder het gebied van algebraische specificatie van abstracte data typen specificaties, zijn specificaties met soorten, opgebouwd via het inductieprincipe, welbekend. Hier gaat het om datatypen zoals natuurlijke getallen, waarheidswaarden (booleans), eindige bomen, eindige lijsten, enz. In recente jaren is een 'duale' specificatiemethode prominent geworden, namelijk de coalgebraische specificatie van oneindige data, ook wel codata genoemd. Hier heerst het principe van coinductie; typische codata zijn 'lazy' natuurlijke getallen, oneindige bomen, of oneindige symboolrijen, ook wel stromen genoemd.

In dit proefschrift ontwikkelen we een algemeen raamwerk dat een uitbreiding is van zowel inductieve als coinductieve specificaties. Nadat we de basisbegrippen hebben opgezet, beschouwen we een bij uitstek wenselijke eigenschap van zulke inductief-coinductieve specificaties: deze moeten *productief* zijn, dat wil zeggen, op de oneindige (coinductieve) delen moet de specificatie gegarandeerd voortdurend 'output' genereren. Deze productiviteitseigenschap is een consequentie van drie secundaire eigenschappen. Ten eerste, *infinitaire normalisatie* (WN) garandeert dat een expressie een (mogelijk oneindige) normaalvorm heeft. Ten tweede, *domein normalisatie* (DN) garandeert dat de normaalvorm voldoet aan de restricties met betrekking tot welke oneindige paden in de termboom toegestaan zijn. Ten derde, *constructor normalisatie* (CN) garandeert dat de normaalvorm is opgebouwd uit uitsluitend constructoren, zonder gedefineerde functiesymbolen. Vervolgens geven we condities voor elk van de drie eigenschappen WN, DN, CN, en ook enkele karakteriseringen. Het eindresultaat in dat we daarmee condities hebben, die garanderen dat een inductief-coinductieve specificatie productief is.

Als toepassing van de zo ontwikkelde theorie, bestuderen we in het laatste hoofdstuk een vrij gecompliceerd oneindig datatype, dat bekend staat als boomordinalen (tree ordinals). Deze zijn belangrijk in de theorie van ordinaalnotaties in de bewijstheorie, een deelgebied van de mathematische logica. Deze studie van boomordinalen is tevens een verkenning van de expressiviteit van het raamwerk van (eerste-orde) termherschrijven, en we tonen aan dat deze expressiviteit zeer groot is: We kunnen ordinalen

beschrijven die veel groter zijn dan het Feferman–Schütte ordinaal dat bekend staat als Γ_0.

We geven nu een beknopte opsomming van de inhoud van de hoofdstukken. Hoofdstuk 0 geeft motivatie, achtergrond en vermeldt wat de bijdrage is van dit proefschrift. Hoofdstuk 1 formaliseert het raamwerk van algoritmische systemen, met een inleiding in begrippen en eigenschappen van infinitair herschrijven, in het bijzonder voor orthogonale herschrijfsystemen. Hoofdstuk 2 bevat technische voorbereidingen, vooral voor gebruik in hoofdstuk 3. Hoofdstuk 3 ontwikkelt een karakterisering en een criterium voor een algoritmisch systeem. Hoofdstuk 4 bestudeert ordinaalsystemen, als voorbeelden van productieve algoritmische systemen. In dit raamwerk implementeren we het kleine Veblen-ordinaal.

Curriculum Vitae

I was born on November 6th, 1980 in Nagoya, Aichi, Japan. In 2003, I graduated with a bachelor degree in mathematical science from Nagoya University. In 2005, I received my master degree in computer science from Kyoto University, under the supervision of prof.dr. Masahito Hasegawa.

In February 2006, I started my PhD study as AIO in the Theoretical Computer Science Section at the Faculty of Science, Vrije Universiteit Amsterdam.

Since March 2007, I have been married with Keiko. In 2008, our son Haruka Isihara was born.